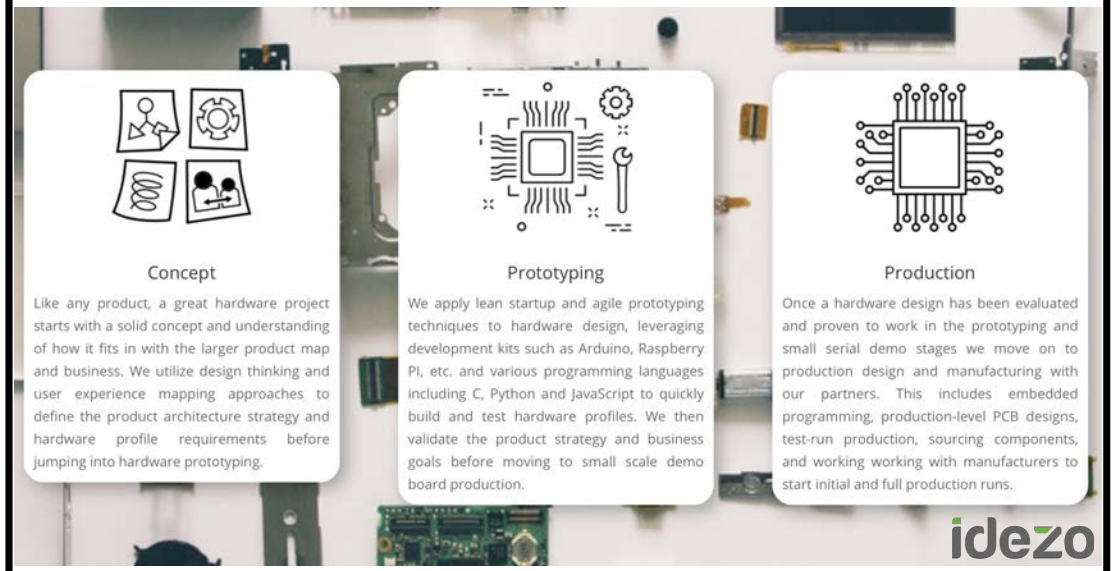
HANDS ON APPROACH TO

Data Science

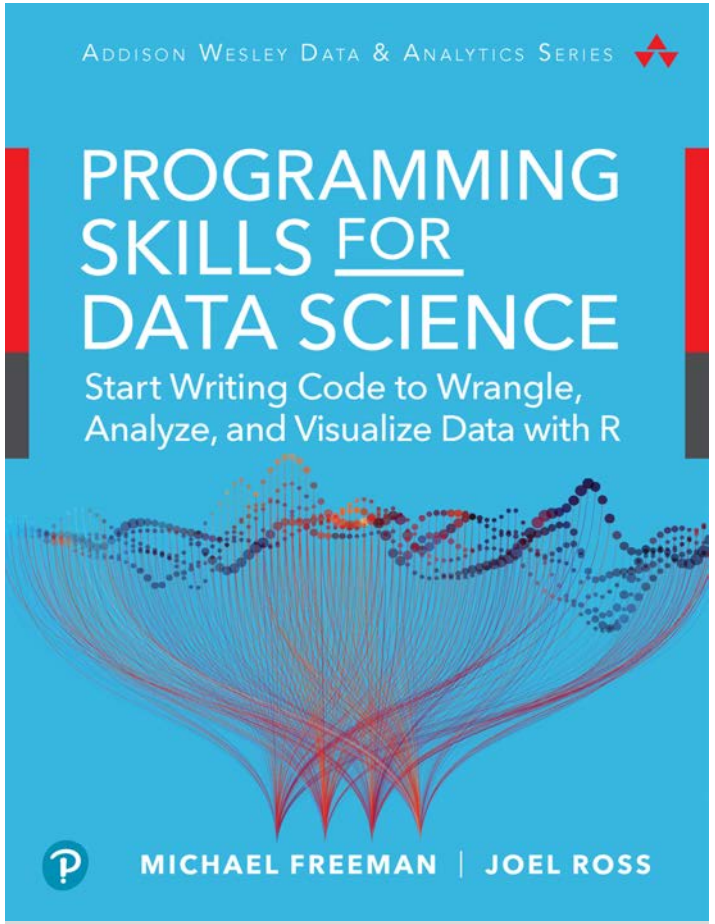
for (the)

IoT



Rob van der Willigen

https://github.com/robvdw/CMIDAT01K-DATA-SCIENCE-for-IOT



http://ptgmedia.pearsoncmg.com/images/9780135133101/samplepages/9780135133101_sample.pdf

Book Structure

The book is divided into six sections, each of which is summarized here.

Part I: Getting Started

This section walks through the steps of downloading and installing necessary software for the rest of the book. More specifically, Chapter 1 details how to install a text editor, Bash terminal, the R interpreter, and the RStudio program. Then, Chapter 2 describes how to use the command line for basic file system navigation.

Part II: Managing Projects

This section walks through the technical basis of project management, including keeping track of the version of your code and producing documentation. Chapter 3 introduces the `git` software to track line-by-line code changes, as well as the corresponding popular code hosting and collaboration service *GitHub*. Chapter 4 then describes how to use Markdown to produce the well-structured and -styled documentation needed for sharing and presenting data.

Part III: Foundational R Skills

This section introduces the R programming language, the primary language used throughout the book. In doing so, it introduces the basic syntax of the language (Chapter 5), describes fundamental programming concepts such as functions (Chapter 6), and introduces the basic data structures of the language: vectors (Chapter 7), and lists (Chapter 8).

Part IV: Data Wrangling

Because the most time-consuming part of data science is often loading, formatting, exploring, and reshaping data, this section of the book provides a deep dive into the best ways to *wrangle* data in R. After introducing techniques and concepts for understanding the structure of real-world data (Chapter 9), the book presents the data structure most commonly used for managing data in R: the data frame (Chapter 10). To better support working with this data, the book then describes two *packages* for programmatically interacting with the data: `dplyr` (Chapter 11), and `tidyr` (Chapter 12). The last two chapters of the section describe how to load data from databases (Chapter 13) and web-based data services with application programming interfaces (APIs) (Chapter 14).

Part V: Data Visualization

This section of the book focuses on the conceptual and technical skills necessary to design and build *visualizations* as part of the data science process. It begins with an overview of data visualization principles (Chapter 15) to guide your choices in designing visualizations. Chapter 16 then describes in granular detail how to use the `ggplot2` visualization package in R. Finally, Chapter 17 explores the use of three additional R packages for producing engaging interactive visualizations.

Part VI: Building and Sharing Applications

As in any domain, data science insights are valuable only if they can be shared with and understood by others. The final section of the book focuses on using two different approaches to creating interactive platforms to share your insights (directly from your R program!). Chapter 18 uses the R Markdown framework to transform analyses into sharable documents and websites. Chapter 19 takes this a step further with the Shiny framework, which allows you to create interactive web applications using R. Chapter 20 then describes approaches for working on collaborative teams of data scientists, and Chapter 21 details how you can further your education beyond this book.

Book Conventions

Throughout the book, you will see computer code appear inline with the text, as well as in distinct blocks. When code appears inline, it will appear in `monospace font`. A distinct code block looks like this:

```
# This is a comment - it describes the code that follows
# The next line of code prints the text "Hello world!"
print("Hello world!")
```

The text in the code blocks is colored to reflect the syntax of the programming language used (typically the R language). Example code blocks often include values that you need to replace. These replacement values appear in `UPPER_CASE_FONT`, with words separated by underscores. For example, if you need to work with a folder of your choosing, you would put the name of your folder where it says `FOLDER_NAME` in the code. Code sections will all include comments: in programming, *comments* are bits of text that are not interpreted as computer instructions—they aren't code, they're just notes about the code! While a computer is able to understand the code, comments are there to help *people* understand it. Tips for writing your own descriptive comments are discussed in Chapter 5.

To guide your reading, we also include five types of special callout notes:

Tip: These boxes provide best practices and shortcuts that can make your life easier.

Fun Fact: These boxes provide interesting background information on a topic.

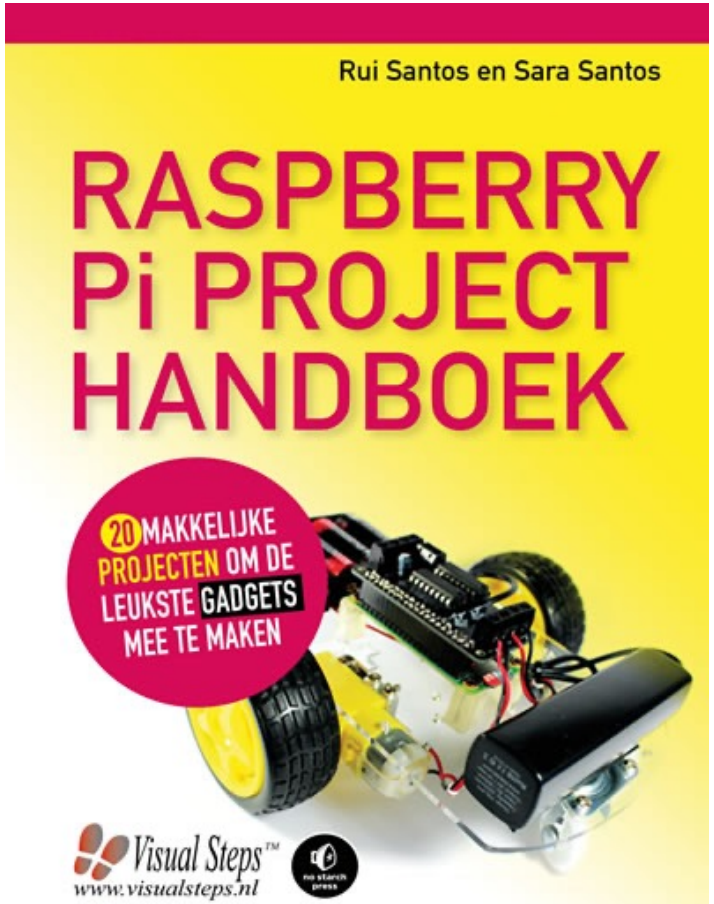
Remember: These boxes reinforce key points that are important to keep in mind.

Caution: These boxes describe common mistakes and explain how to avoid them.

Going Further: These boxes suggest resources for expanding your knowledge beyond this text.

Throughout the text there are instructions for using specific keyboard keys. These are included in the text in `lowercase monospace font`. When multiple keys need to be pressed at the same time, they are separated by a plus sign (`+`). For example, if you needed to press the Command and "c" keys at the same time, it would appear as `Cmd+c`.

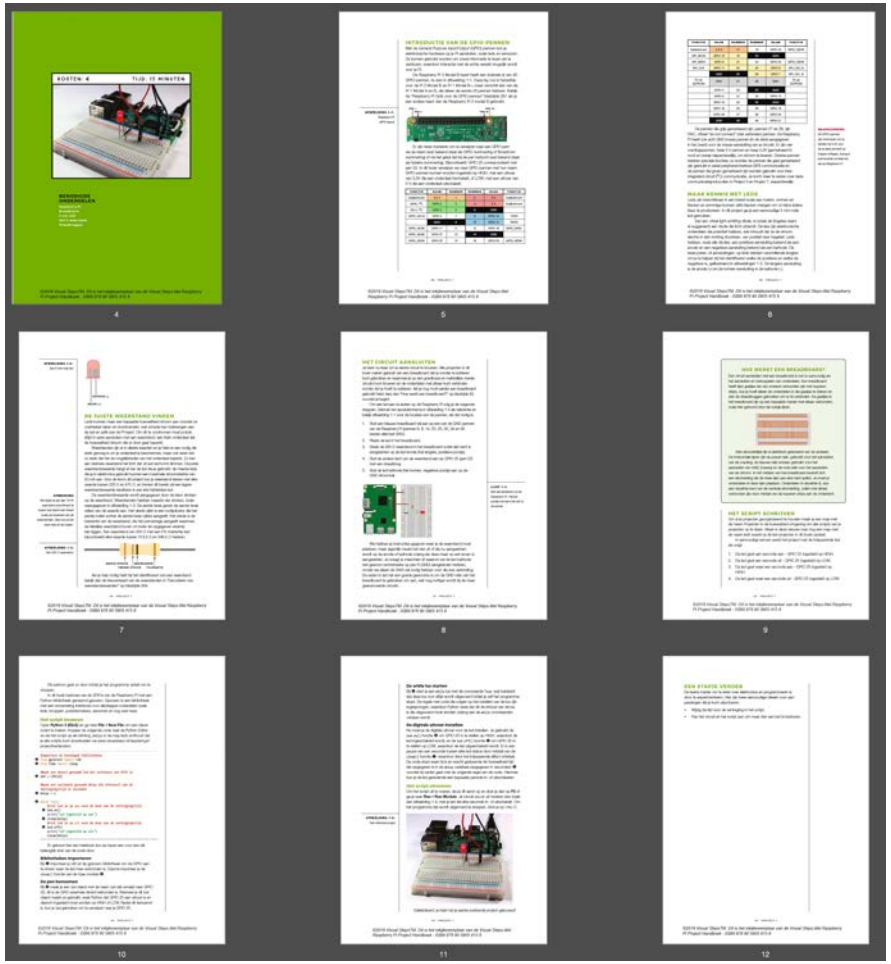
Whenever the `cmd` key is used, Windows users should instead use the Control (`ctrl`) key.



https://www.visualsteps.nl/raspberrypi/
https://www.visualsteps.nl/inkijkexemplaar/957.pdf

1 EEN LED LATEN KNIPPEREN

IN DIT EERSTE PROJECT GA JE EEN LED AANSLUITEN OP JE PI EN DEZE LATEN KNIPPEREN MET EEN PYTHON SCRIPT. HET LEREN HOE JE EEN LED LAAT KNIPPEREN MET DE GPIO-PENNEN IS EEN BELANGRIJKE STAP IN HET LEREN OVER JE PI. ALS JE EENMAAL WEET HOE JE EEN LED KUNT BESTUREN, KUN JE BIJNA ELKE UITVOER BESTUREN, OF HET NOU EEN MOTOR, EEN LAMP OF ZELFS EEN BROODROOSTER IS.



Learning O'Reilly is een Engelstalig online leerplatform gericht op ICT vaardigheden. Het biedt tienduizenden e-books en vele video's, case studies en "learning paths", waarbij je zelf je voortgang kunt monitoren.

Toegang

Via de website

De Learning O'Reilly website is gekoppeld aan de Hogeschool Rotterdam login. Kom je toch een O'Reilly inlogscherm tegen, vul dan alleen je @hr.nl e-mailadres in en klik op **Sign In with Single Sign On**. Krijg je een blanco pagina te zien? Schakel eventuele adblockers uit!

Via de O'Reilly app

Leer wat je wil, waar je wil. Met de O'Reilly app download je e-books en ga je verder met een playlist waar je op een ander apparaat gebleven was.

1 Google play Download on the App Store

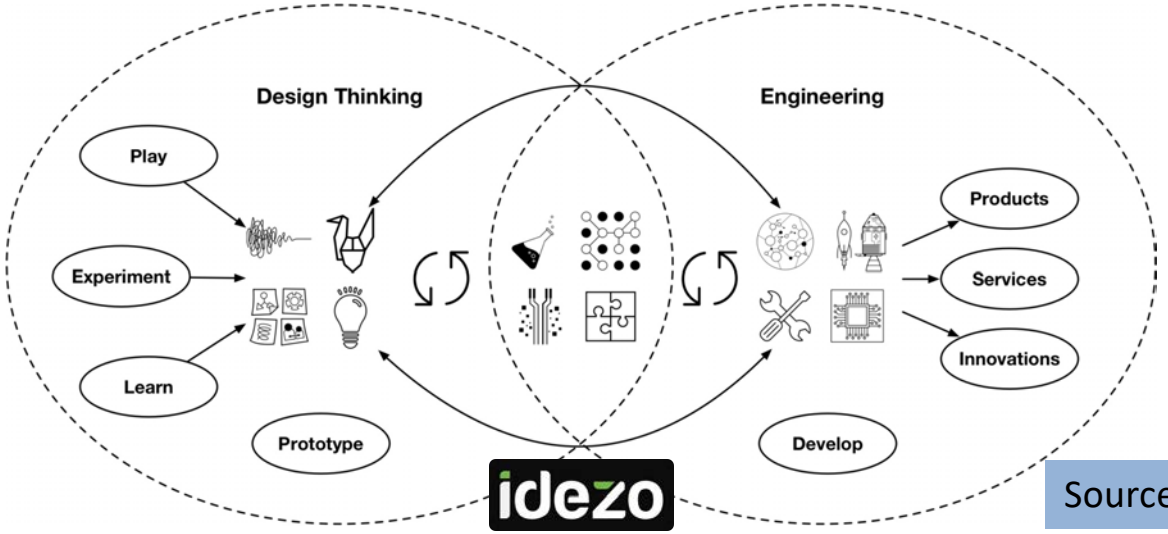
2 Gebruik je @hr.nl e-mailadres om in te loggen.

Inhoud

- Meer dan 8000 instructievideo's en 42.000 (hoofdzakelijk Engelstalige) e-books.

https://www.hogeschoolrotterdam.nl/voorlichting/voorzieningen/mediatheek/informatiebronnen/databases/mtdb/learning_oreilly/184698/

HANDS ON APPROACH TO DATA SCIENCE for (the) IoT



Source: https://idezo.ch

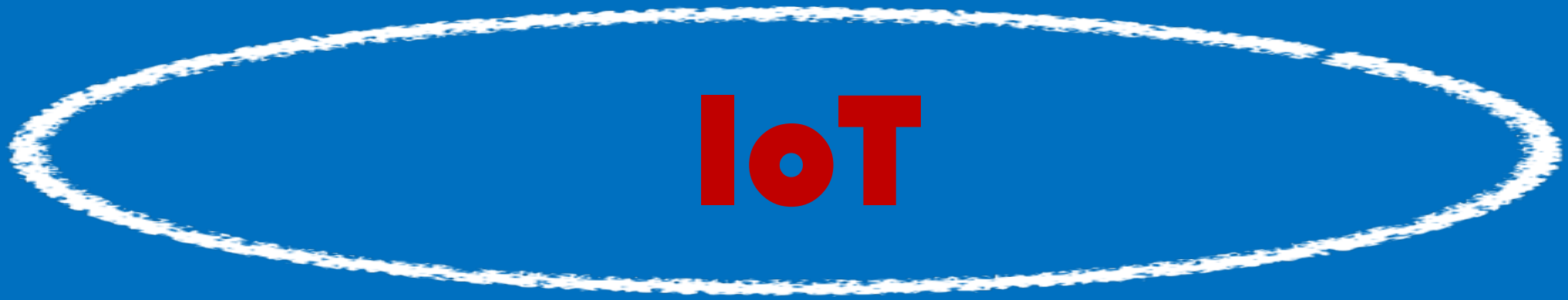
This Data Science Course was developed for Hogeschool Rotterdam **(Rotterdam University of Applied Sciences, RUAS) through the Program for AI & ethics (SPAiCE)**.
If you find errors or omissions, please contact the author, Rob van der Willigen, at r.f.van.der.willigen@hr.nl. Materials of this course and code examples used will become available at:
https://github.com/robvdw/CMIDAT01K-DATA-SCIENCE-for-IOT

Course Setup

HOGESCHOOL ROTTERDAM

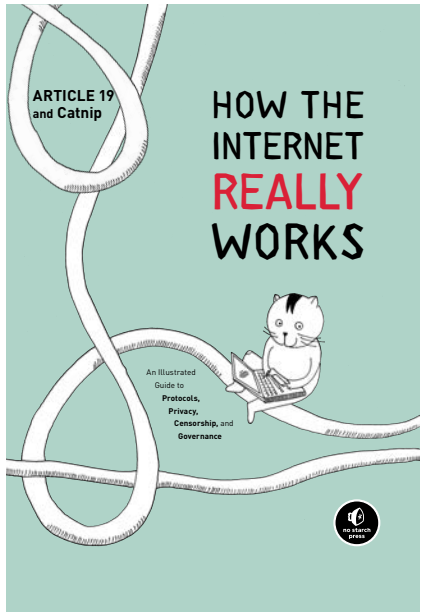
Lesson 01:	week 02	**Discovering the IoT Data Science Domain**
Lesson 02:	week 03	**Defining project requirements**
Lesson 03:	week 04	**Learn to write code**
Lesson 04:	week 05	**Data Science: How to start your own IoT Project**
Lesson 05:	week 06	**IoT Platforms & MiddleWare**
Lesson 06:	week 07	**Core IoT Concepts + Code-testing via IoT** middelware
Lesson 07:	week 08	**Explaining Grading + Summary + Q & A**

Week 08:	**FEEDBACK**
Week 10:	**Submit your IoT-Project via LMS**



HOGESCHOOL ROTTERDAM

https://medium.com/@smartrac/the-deep-web-the-dark-web-and-simple-things-2e601ec980ac



The Layers of the Internet

The internet is made of several layers, one on top of the other and interacting with each other. Now that we have learned about the concrete functions and components of the internet, it's time to conceptualize the internet as a whole system, including the people and institutions that operate it. Each layer operates on the layer below it and serves the layer above it.

Social Layer

The most relatable layer of the internet is made up of the entities that use it and the human relationships that govern it.


citizens


companies


nonprofit organizations

Content Layer

The content layer, what data is accessible and available over the internet, is perhaps the most recognizable for users.


websites (news sites, social media, blogs)


entertainment

Application Layer

Applications are the ways content is served.

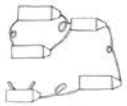

web-based platforms


email

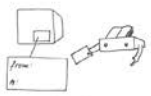

instant messaging

Logical Layer

The logic of the interoperable internet, or its standard protocols, support the connections between devices and the applications running on them.


networking


routing


addressing (Domain Name System)

Infrastructural Layer

Internet infrastructure is the material basis of the IP network, or the physical components across which the logical layer can send information from one place to another.

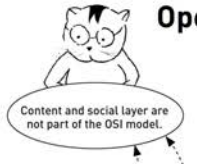

computers (servers, personal computers, mobile devices)


internet of things (IoT)


routers (gateways, switches)


data centers

Open Systems Interconnection (OSI) Model



In 1984, the International Organization for Standardization (ISO) published the **Open Systems Interconnection (OSI) model**, a conceptual model that characterized and standardized the technical communication of a telecommunication or computing system. As a concept, in which any layer can take for granted underlying layers, it enables the interoperability of the internet as a diverse communication system across its governance space.


governments


virtual Reality (VR)


voice over IP


traffic controls


telecommunication cables, Wireless networks, Satellite networks

Layer 7: Application
This layer is the layer with which the user directly interacts using client applications, for example a web browser.

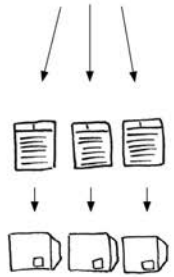
Layer 6: Presentation
This layer translates and transmits data between applications, such as text encoding or audio/visual compression.

Application Layer

Layer 5: Session
This layer controls the connections between computers by opening the connection, or session, and closing it, according to various session layer protocols.

Layer 4: Transport
This layer ensures the reliable transmission of packets, grouped in so-called messages, segments (TCP), or datagrams (UDP).

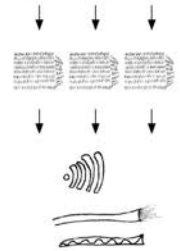
Layer 3: Network
This layer defines an addressing scheme and how packets are routed over the network.



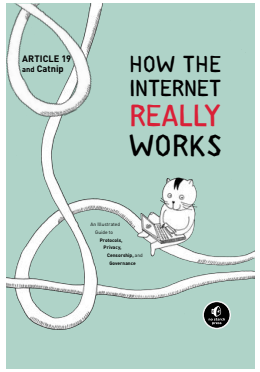
Logical Layer

Layer 2: Data-Link
This layer defines the transmission of data frames between two nodes directly connected in the physical layer.

Layer 1: Physical
This layer defines electrical and physical specifications of the data connection. At this layer, physical media (electrical cable, optical glass fiber, radio frequency spectrum) send and receive raw bit streams. Network adapters, repeaters, and modems operate only at this low level.



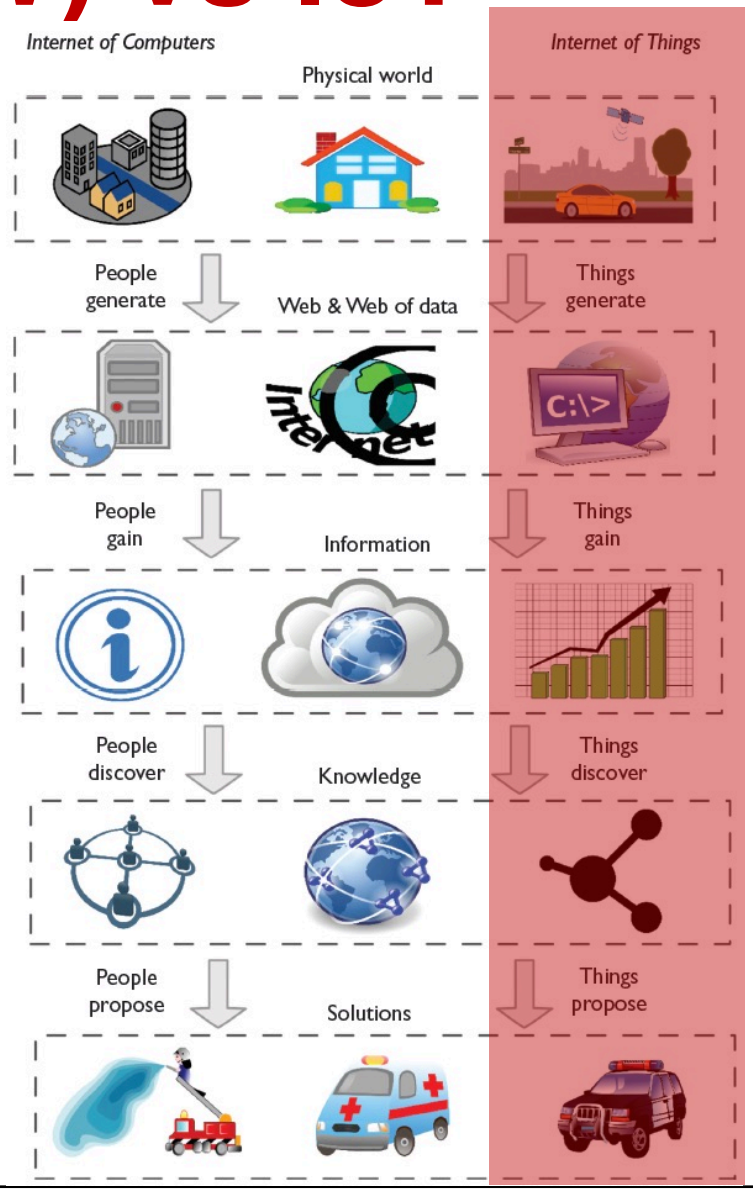
Infrastructural Layer



World Wide Web (WWW) vs IoT

WWW:
In the **Internet of Computers (WWW)**, the main **data producers** and consumers are **human beings**.

IoT:
The main actors become **things**, where things are the majority of **data producers** and consumers.

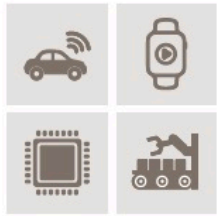


Qin, Y., Sheng, Q. Z., & Curry, E. (2015). Matching over linked data streams in the internet of things. *IEEE Internet Computing*, *19*(3), 21-27.

3 Pillars of the IoT

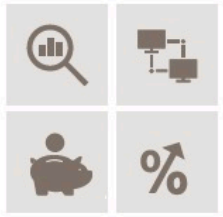
1 things

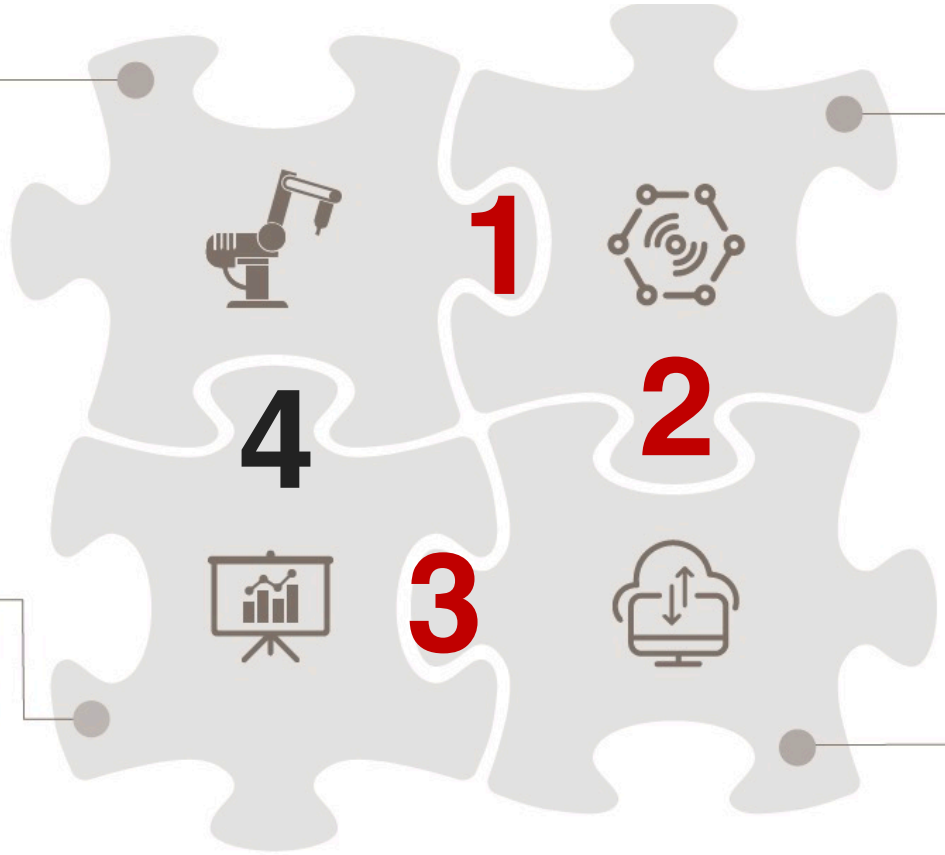
- Sensors
- Actuators
- Machines



4 Data Products

- Outcome driven
- Product as a service
- Leverage existing investment





2 Connectivity

- Network
- Protocol
- Security

3 People + Processes

- Storage
- Analytics
- Visualization
- Integration

IoT User-Types

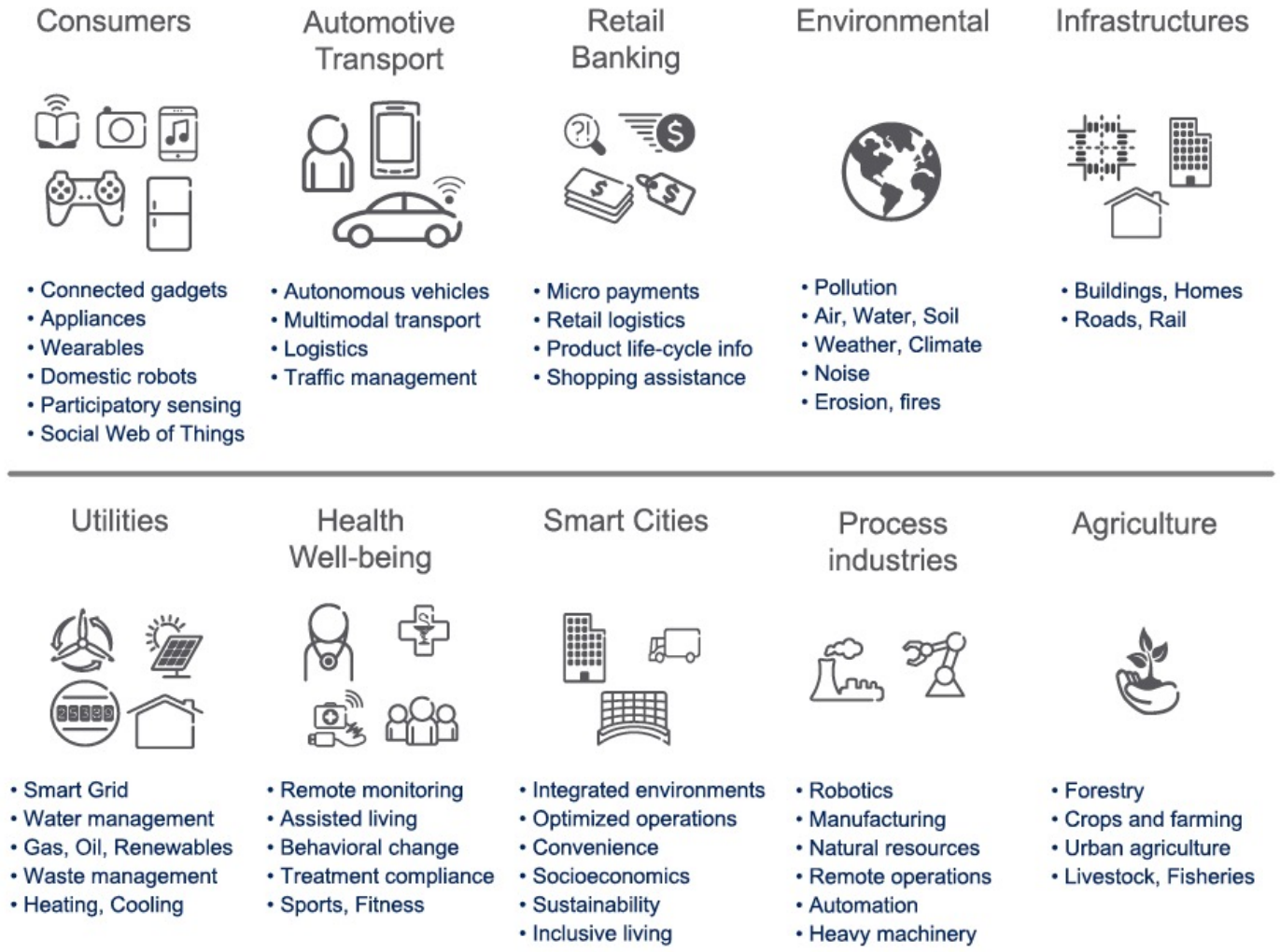
Hobbyists

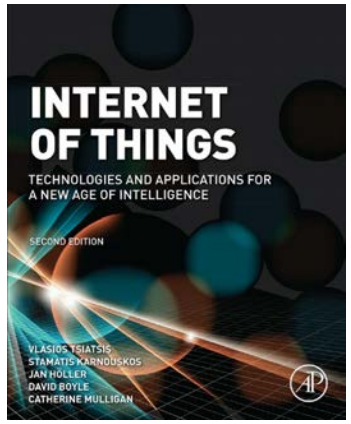
Consumer

Industrial

Industry

IoT Application Domains





IoT

HOGESCHOOL ROTTERDAM

Smart Systems and the Internet of Things are driven by a combination of:

http://postscapes.com/what-exactly-is-the-internet-of-things-infographic

1 things

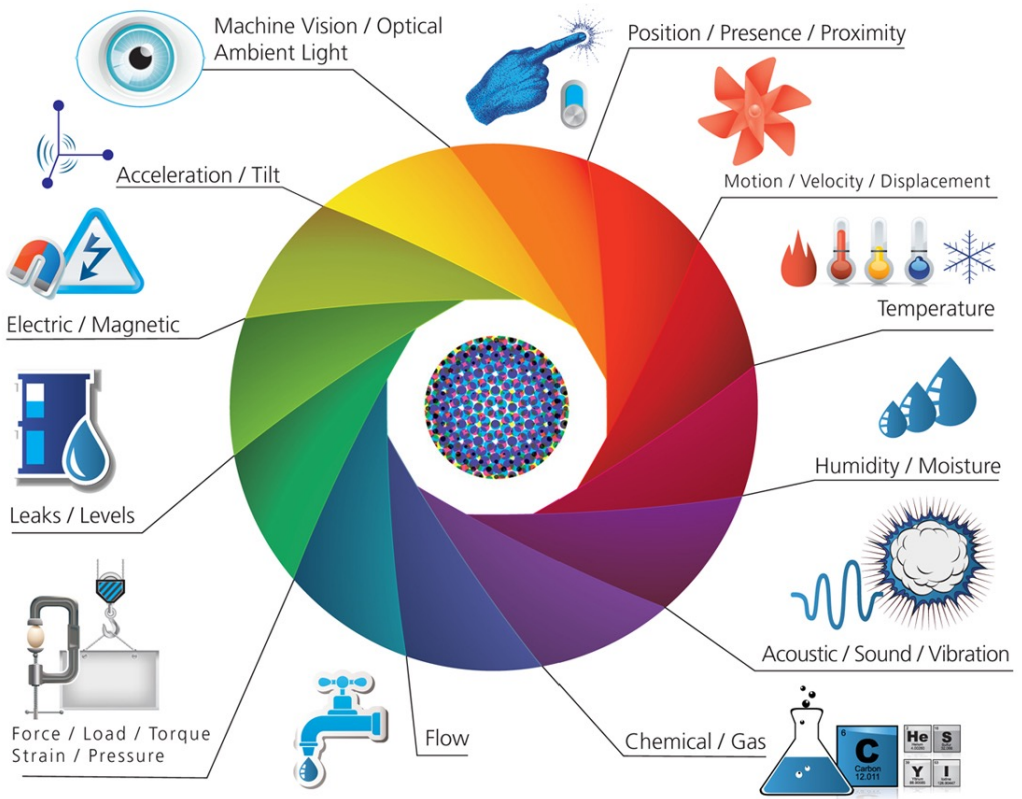
ACTUATORS

An actuator is a component of a machine that is responsible for moving or controlling a mechanism or system. An actuator requires a control signal and a source of energy.

The control signal is relatively low energy and may be electric voltage or current, pneumatic or hydraulic pressure, or even human power.

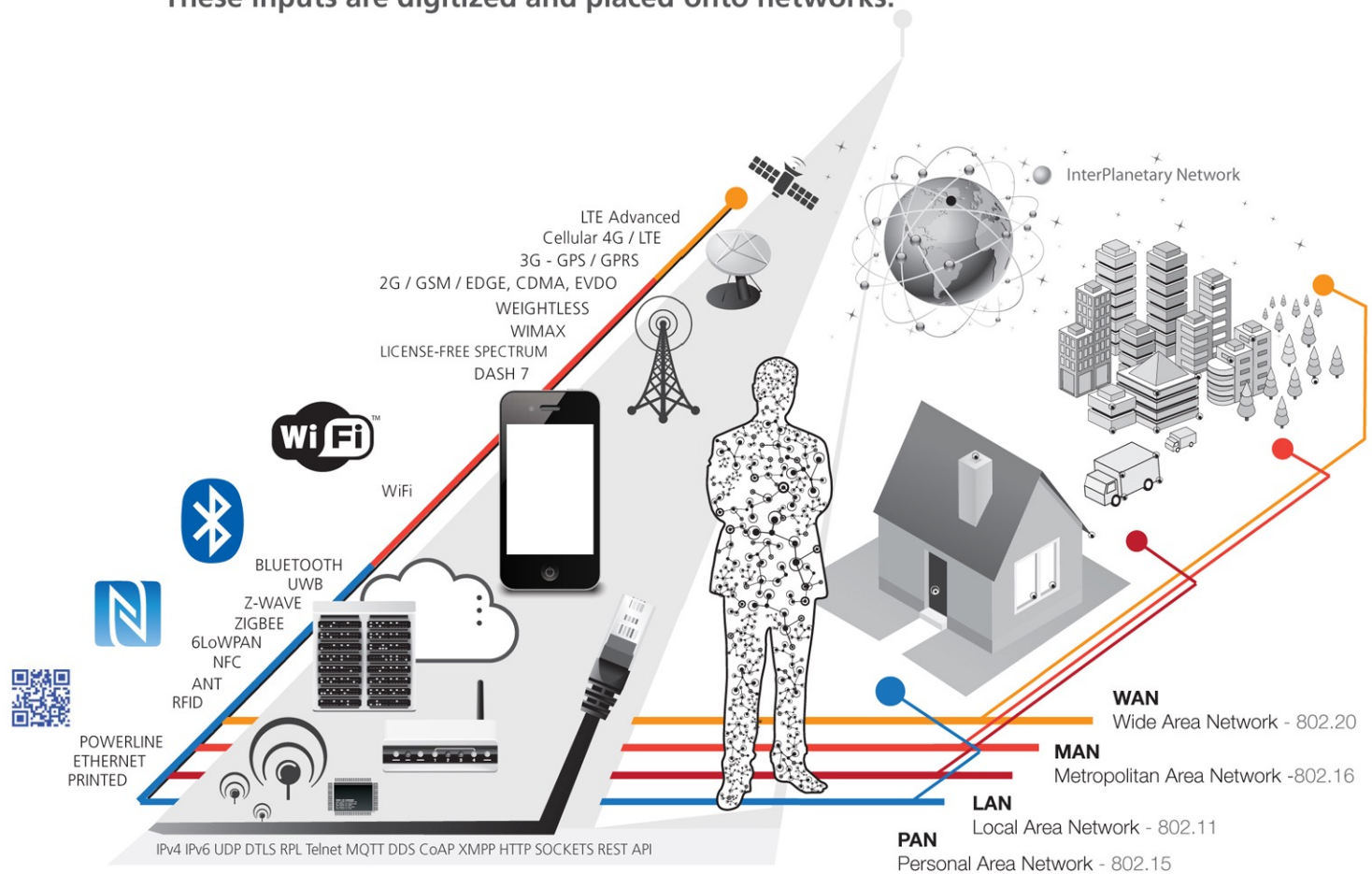
1 SENSORS & *ACTUATORS*

We are giving our world a digital nervous system. Location data using GPS sensors. Eyes and ears using cameras and microphones, along with sensory organs that can measure everything from temperature to pressure changes.



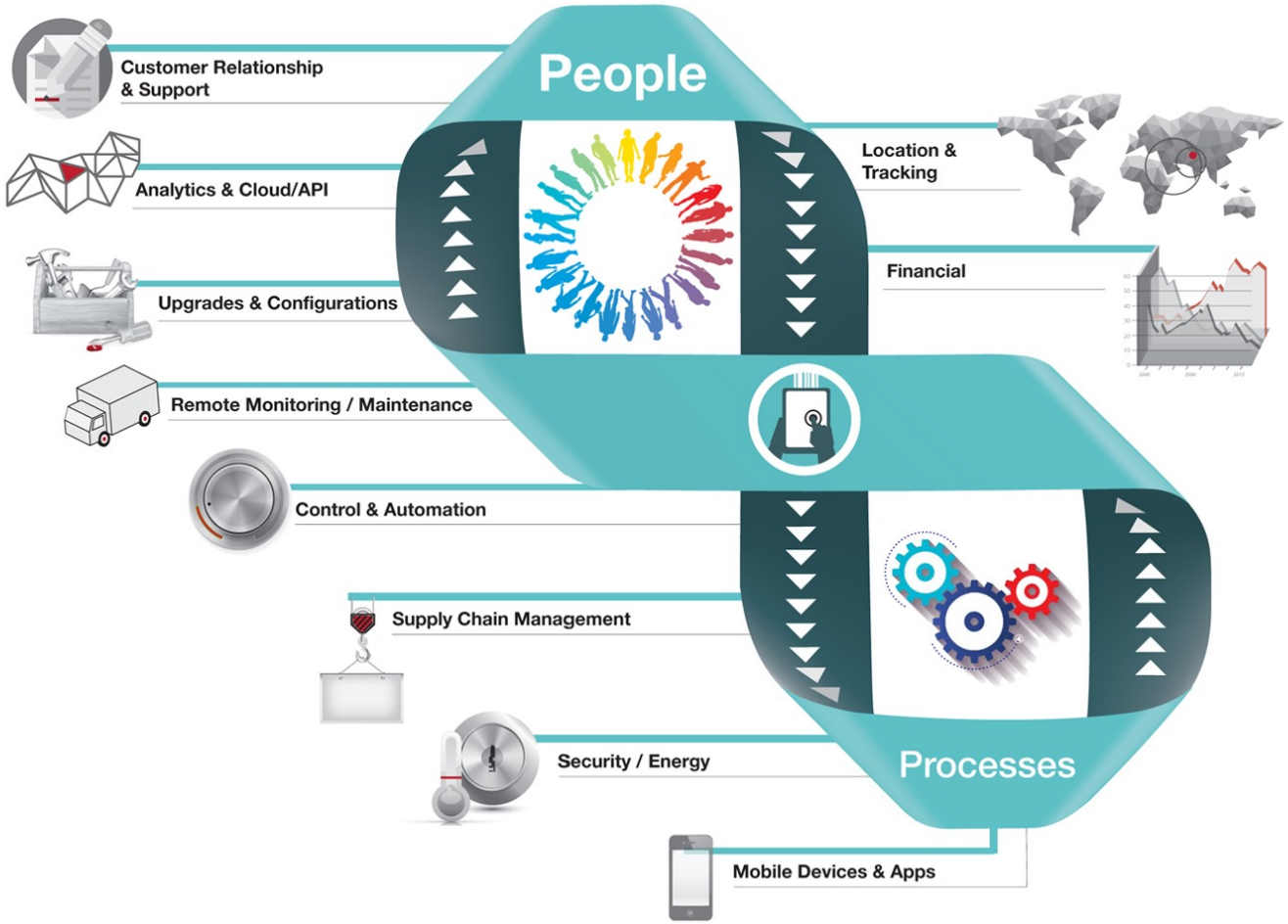
2 CONNECTIVITY

These inputs are digitized and placed onto networks.



3 PEOPLE & *PROCESSES*

These networked inputs can then be combined into bi-directional systems that integrate data, people, processes and systems for better decision making.

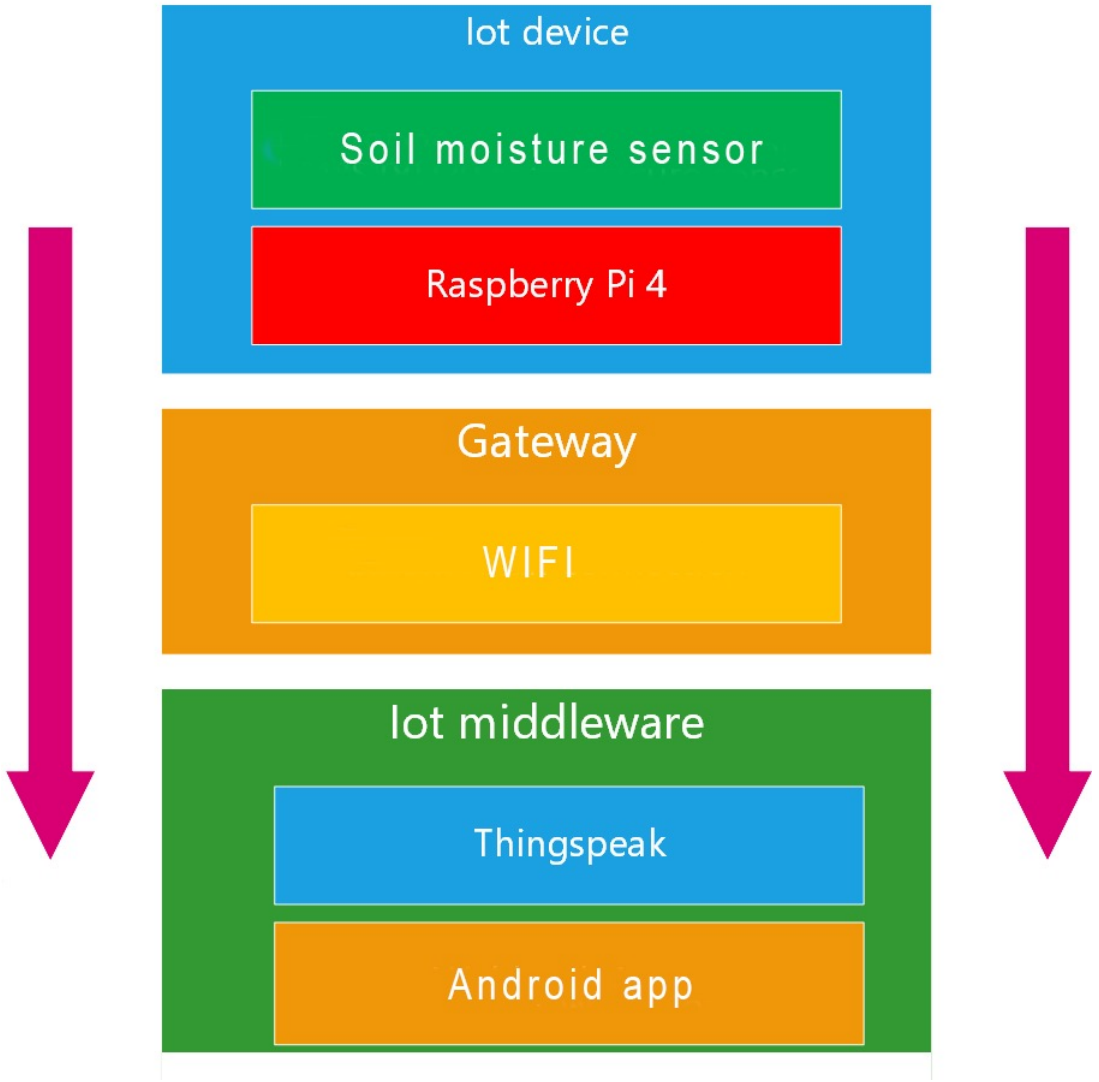


IoT Data Pipeline

Collect

Transform

Store & Analyse



Technologies enabling IoT

WIRELESS TECHNOLOGIES

802.11, 802.15.4, Zigbee, Bluetooth, BLE, CDMA, GSM, LTE

NETWORK TECHNOLOGIES

Ethernet, CAN, rs485/rs422/rs232, 1-wire, i2c, SPI, ModBus/MudBusRTU, IPMI, iSCSI

WEB SERVICE TECHNOLOGIES

SOAP, REST, WSDL, XML, JSON, UDDI, WebSockets

TCP/IP PROTOCOLS

Sockets, IPv4, IPv6, TCP, UDP, ICMP, QOS, SNMP, IMAP, POP3, IPMI, etc.

MOBILE TECHNOLOGIES

Android SDK, Qt, iOS SDK, Objective C, Java, Swift

PROTOCOL TECHNOLOGIES

HTTP, JMS, AMQP, D-Bus

AUTHORIZATION & AUTHENTICATION TECHNOLOGIES

Oauth2, PAM, LDAP

PLATFORMS AND CPU ARCHITECTURES

ARM, X86, PowerPC, AVR, PIC

PROGRAM / SCRIPTING LANGUAGES

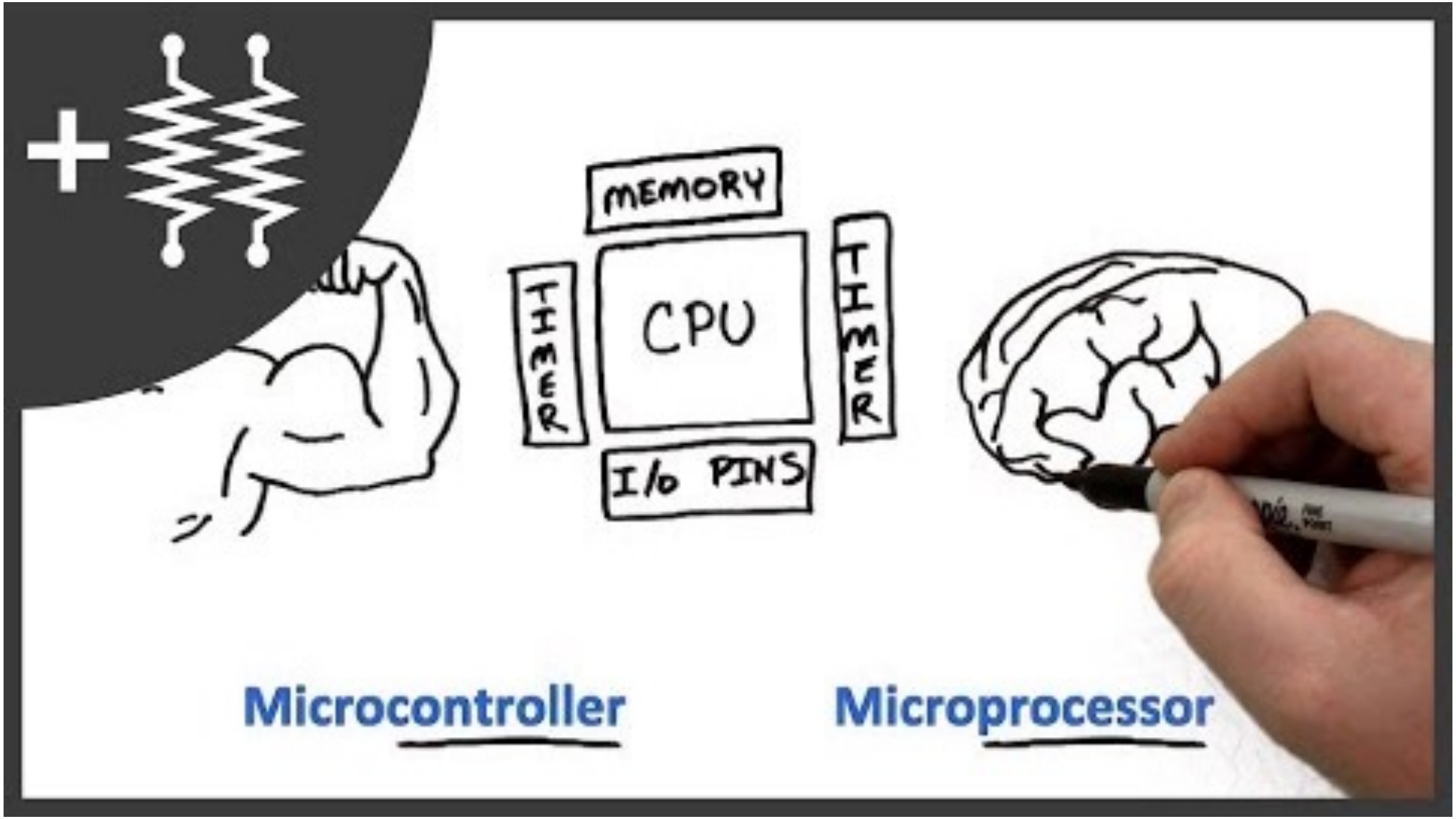
Java, C/C++, C#, JavaScript, Ruby, Groovy, Python, Tcl/Tk, ASM, Bash

https://doi.org/10.15779/Z38WW0V

https://doi.org/10.1016/j.eswa.2019.05.014

Getting to Know Your IoT Device

HOGESCHOOL ROTTERDAM



https://www.youtube.com/watch?v=7vhvnaWUZjE

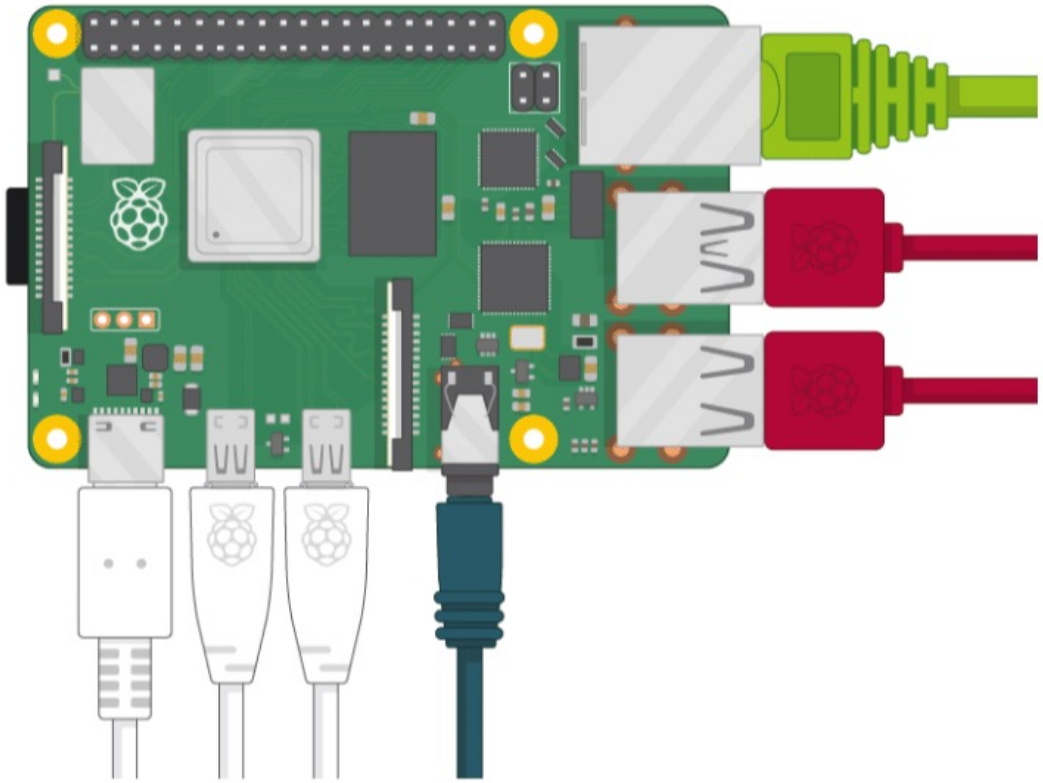
Raspberry Pi

https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up

- Introduction
- What you will need
- Set up your SD card
- Connect your Pi
- Start up your Pi
- Finish the setup
- Where to find help
- What next?

Introduction

Here you'll learn about your Raspberry Pi, what things you need to use it, and how to set it up.

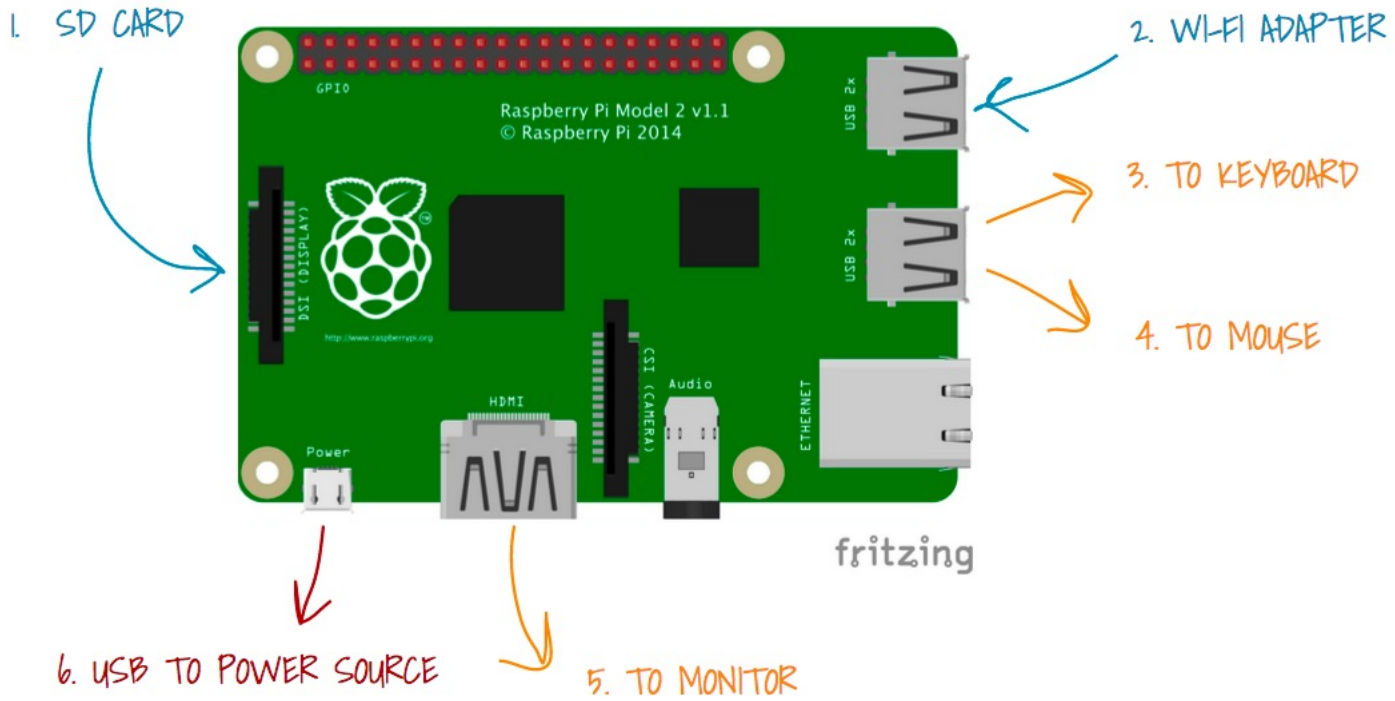


What is a Raspberry Pi?

The Raspberry Pi is a low cost, **credit-card sized computer** that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python. It's capable of doing everything you'd expect a desktop computer to do, from browsing the internet and playing high-definition video, to making spreadsheets, word-processing, and playing games.

What's more, the Raspberry Pi has the ability to interact with the outside world, and has been used in a wide array of digital maker projects, from music machines and parent detectors to weather stations and tweeting birdhouses with infra-red cameras. We want to see the Raspberry Pi being used by kids all over the world to learn to program and understand how computers work.

Main Components Raspberry Pi



https://www.pubnub.com/blog/internet-of-things-101-getting-started-w-raspberry-pi/

Keyboard + Mouse + HDMI

https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up

HOGESCHOOL ROTTERDAM

A keyboard and a mouse

To start using your Raspberry Pi, you need a USB keyboard and a USB mouse.

Once you've set up your Raspberry Pi, you can use a Bluetooth keyboard and mouse, but you'll need a USB keyboard and mouse for the first setup.

A TV or computer screen

To view the Raspberry Pi OS desktop environment, you need a screen, and a cable to link the screen and your Raspberry Pi. The screen can be a TV or a computer monitor. If the screen has built-in speakers, Raspberry Pi is able to use these to play sound.

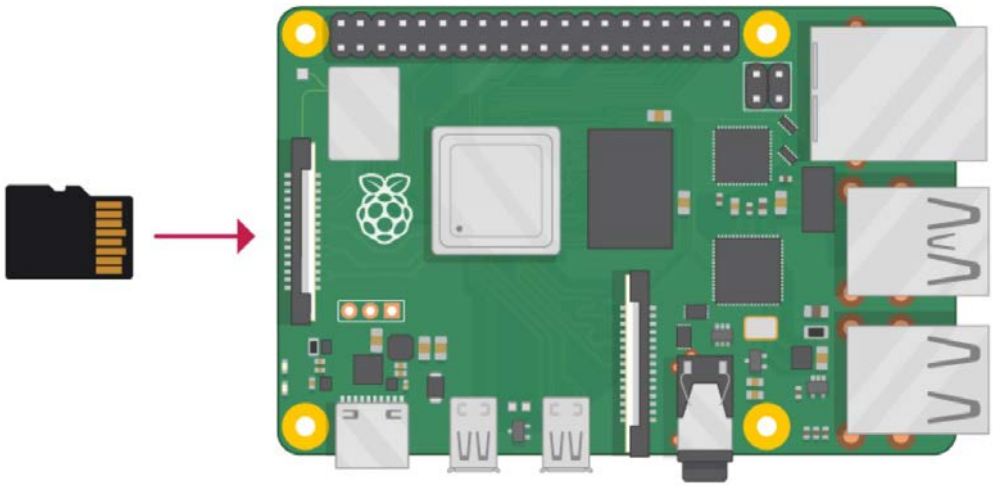
HDMI

Your Raspberry Pi has an HDMI output port that is compatible with the HDMI port of most modern TVs and computer monitors. Many computer monitors may also have DVI or VGA ports.

Micro SD Card

https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up

Your Raspberry Pi needs an SD card to store all its files and the Raspberry Pi OS operating system.



You need a microSD card with a capacity of **at least 8GB**.

If you have an SD card that doesn't have the Raspberry Pi OS operating system on it yet, or if you want to reset your Raspberry Pi, you can easily install Raspberry Pi OS yourself. To do so, you need a computer that has an SD card port — most laptop and desktop computers have one.

The Raspberry Pi OS operating system via the Raspberry Pi Imager

Using the Raspberry Pi Imager is the easiest way to install Raspberry Pi OS on your SD card.

Note: More advanced users looking to install a particular operating system should use this guide to **installing operating system images**.

Etching Raspbian onto a SD card

Set up your SD card

If you have an SD card that doesn't have the Raspbian operating system on it yet, or if you want to reset your Raspberry Pi, you can easily install Raspbian yourself. To do so, you need a computer that has an SD card port — most laptop and desktop computers have one.

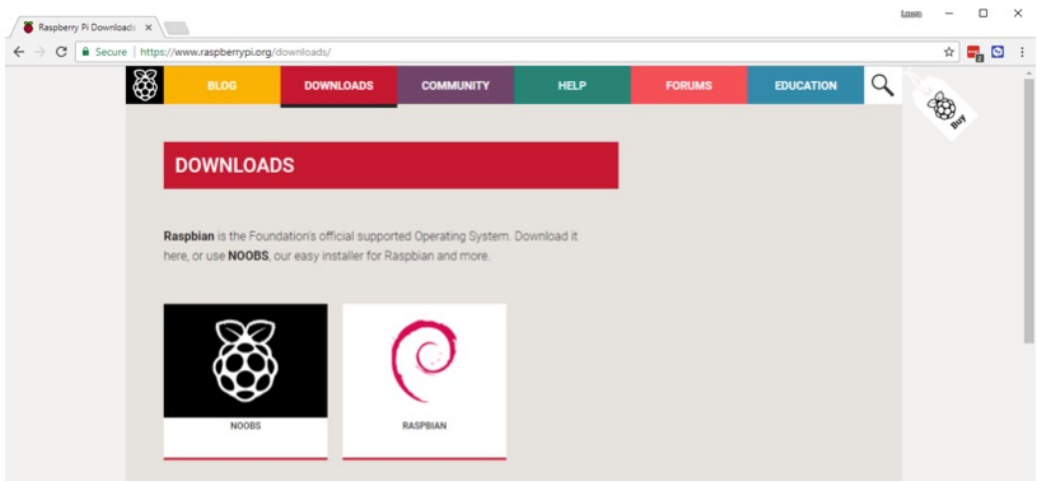
The Raspbian operating system via NOOBS

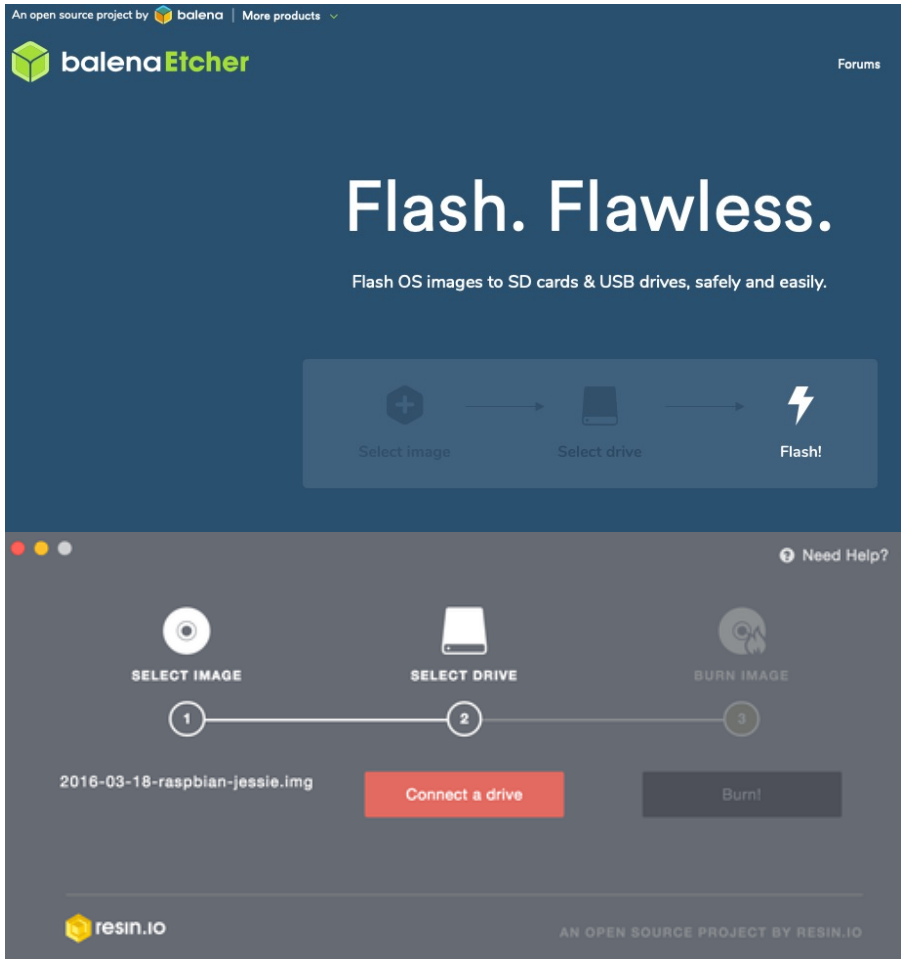
Using the NOOBS software is the easiest way to install Raspbian on your SD card.

Note: more advanced users looking to install a particular operating system should use this guide to installing operating system images.

Download NOOBS

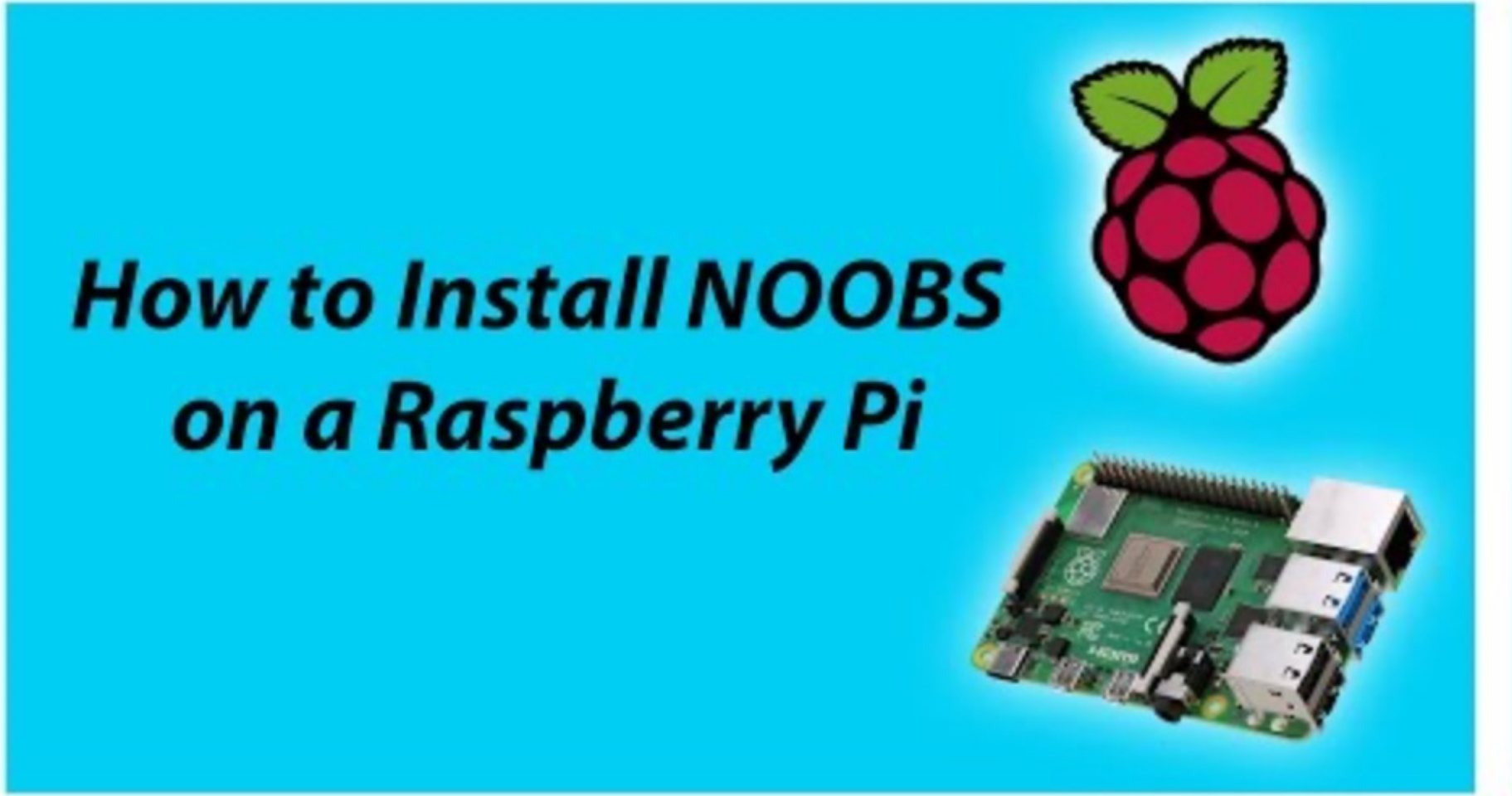
- Visit the Raspberry Pi downloads page.





https://www.balena.io/etcher/

How to Install NOOBS on Raspberry Pi



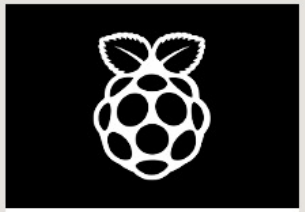
https://www.youtube.com/watch?v=TlQxaEdyBgM

Raspberry Pi Operating Systems

HOGESCHOOL ROTTERDAM

Downloads

Raspbian is our official operating system for **all** models of the Raspberry Pi. Download it here, or use **NOOBS**, our easy installer for Raspbian and more.



NOOBS

Raspbian

Raspberry Pi Desktop (for PC and Mac)

Debian with Raspberry Pi Desktop is the Foundation's operating system for PC and Mac. You can create a live disc, run it in a virtual machine, or even install it on your computer.

Raspberry Pi Desktop

Third Party Operating System Images

Third-party operating system images for Raspberry Pi are also available:

Ubuntu MATE

Ubuntu Core

Ubuntu Server

Windows 10 IoT Core

OSMC

LibreELEC

Mozilla WebThings

PiNet

Raspberry Pi Operating Systems

01/25/2019 • 12 minutes to read

May 2017

Volume 32 Number 5

[Internet of Things]

Working with Raspberry Pi and Windows 10

By Bruno Sonnino

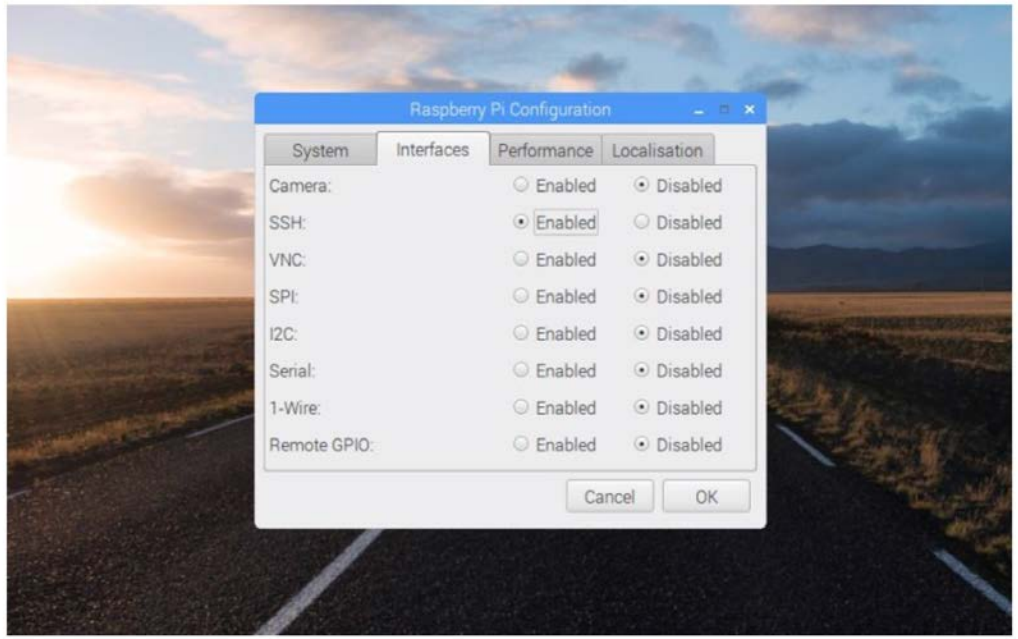
Although I've been working with software for a long time, I've never interacted directly with hardware. I've developed a lot of software that works near the hardware, but I've never worked with a physical board where I have complete control of what's being done. Therefore, when I had the opportunity to work with the Raspberry Pi, especially using Windows 10 and Visual Studio, I jumped at the opportunity.

The Raspberry Pi, in versions 2 and 3, can use Windows 10 as its OS (though it's not the full version, it lets you execute Universal Windows Platform [UWP] apps to control its devices). This is a cheap computer—you can get one for less than $35—and it's powerful. The Raspberry Pi 3 has a Quad-Core, 64-bit ARM processor, HDMI video, Ethernet and Wi-Fi networking, Bluetooth, and four USB ports. You can definitely do many things with it.

https://docs.microsoft.com/en-us/archive/msdn-magazine/2017/may/internet-of-things-working-with-raspberry-pi-and-windows-10

Raspberry Pi 3A+

Connect remotely



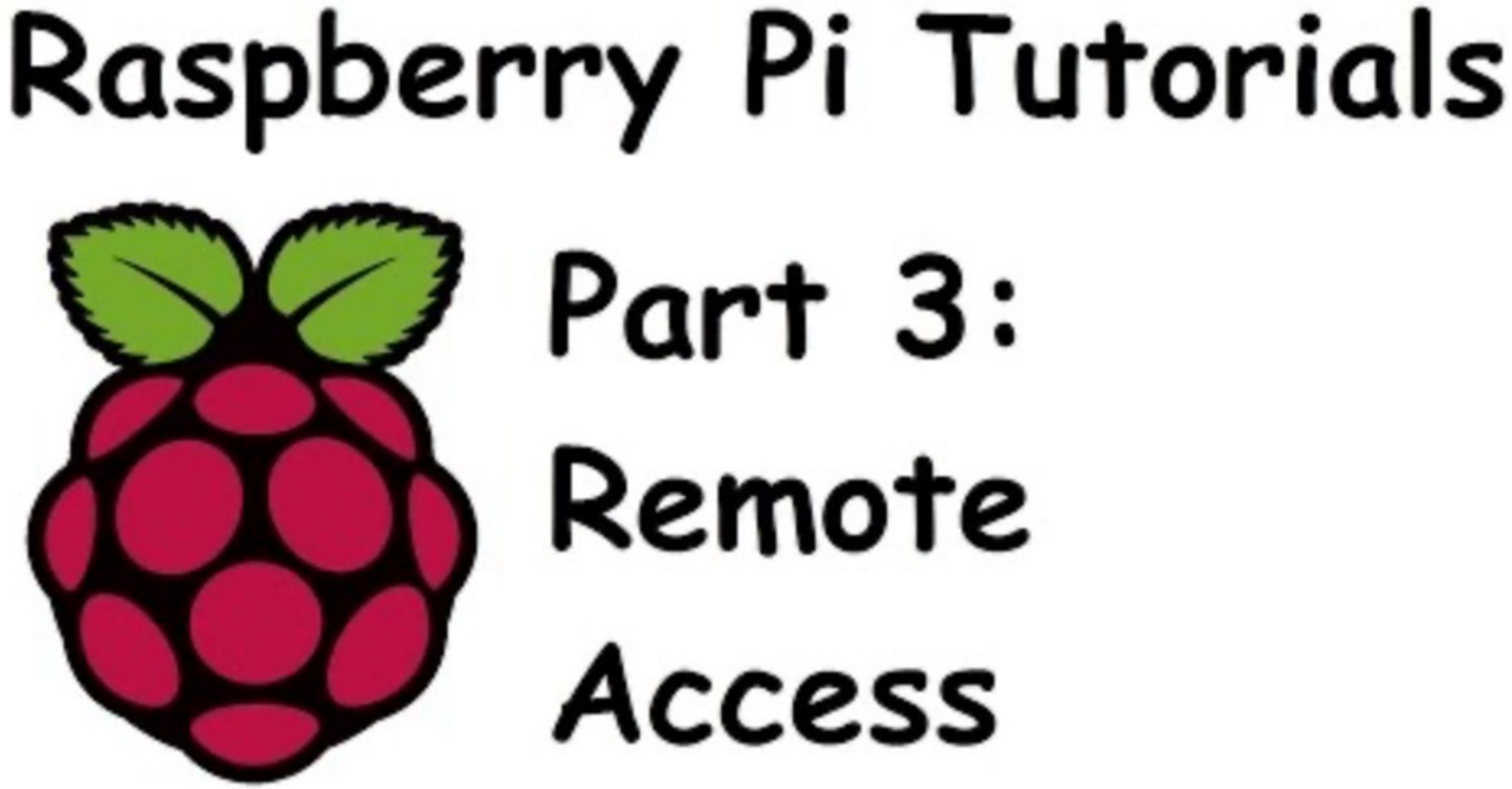
Remotely control your Raspberry Pi from a PC, Linux, or Mac computer and transfer files using SSH.

https://core-electronics.com.au/tutorials/how-to-connect-to-raspberry-pi-3A-plus-remotely-using-ssh.html

https://magpi.raspberrypi.org/articles/vnc-raspberry-pi

Raspberry Pi 3A+

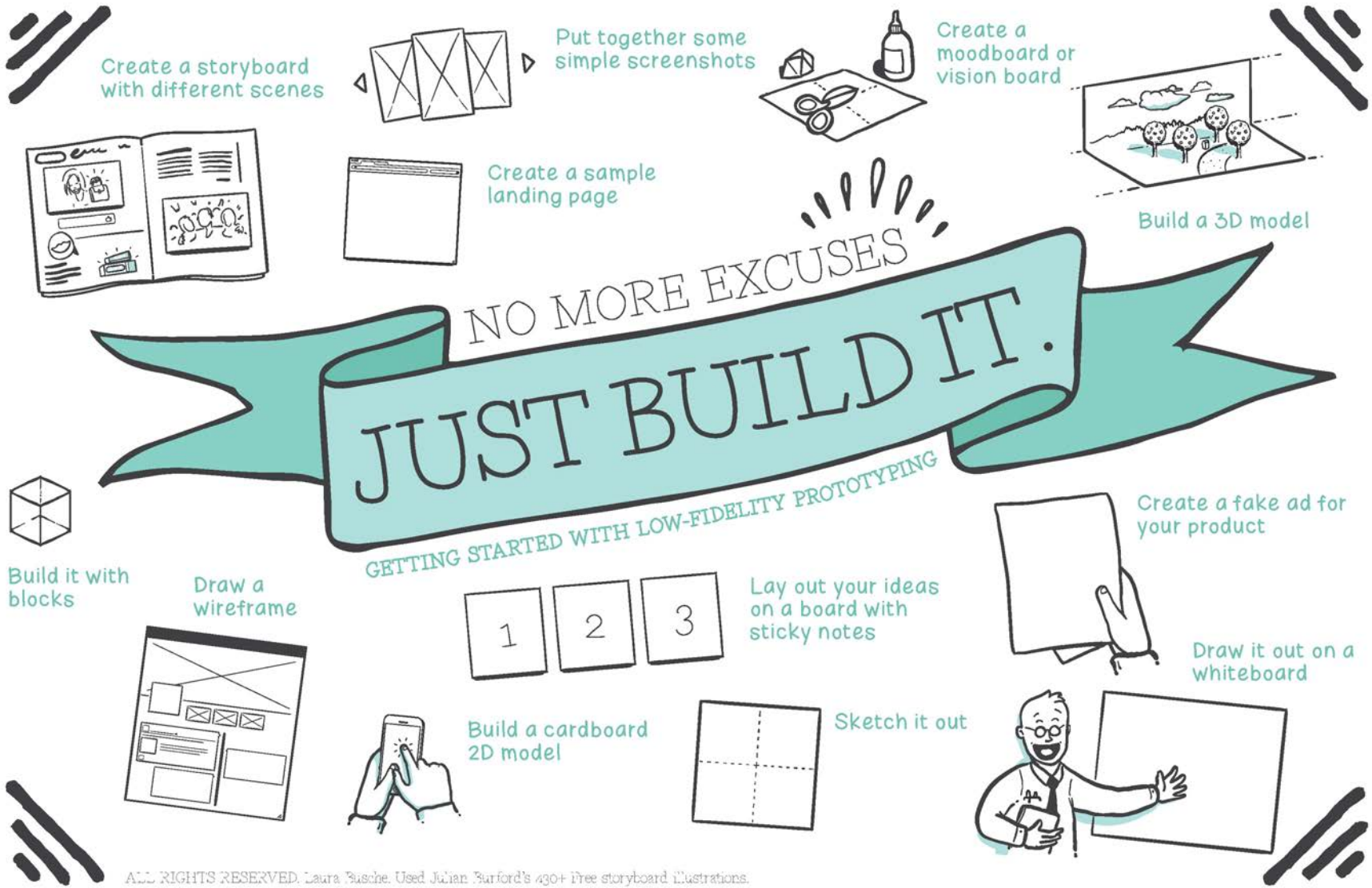
Connect remotely



https://www.youtube.com/watch?v=IDqQIDL3LKg

Rapid Prototyping

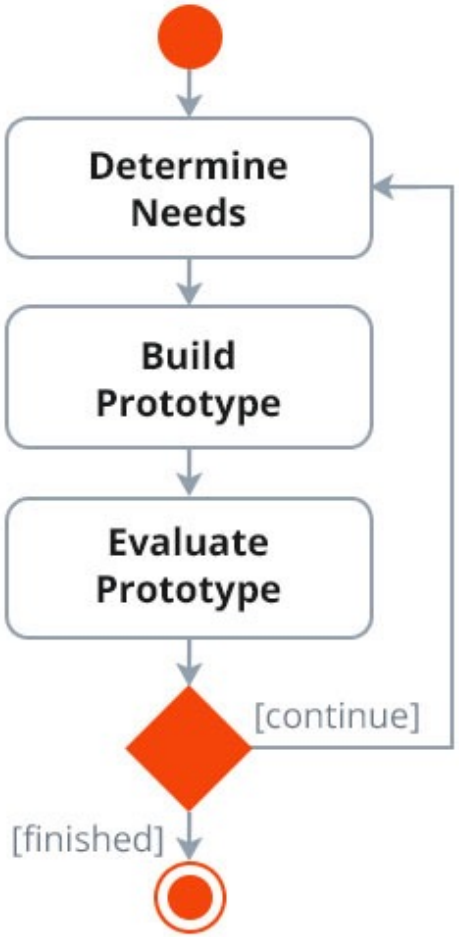
HOGESCHOOL ROTTERDAM


Create a storyboard with different scenes
Put together some simple screenshots
Create a moodboard or vision board
Create a sample landing page
Build a 3D model
NO MORE EXCUSES
JUST BUILD IT.
GETTING STARTED WITH LOW-FIDELITY PROTOTYPING
Build it with blocks
Draw a wireframe
1
2
3
Lay out your ideas on a board with sticky notes
Create a fake ad for your product
Draw it out on a whiteboard
Build a cardboard 2D model
Sketch it out
ALL RIGHTS RESERVED. Laura Busche. Used Julian Burford's 430+ Free storyboard Illustrations.

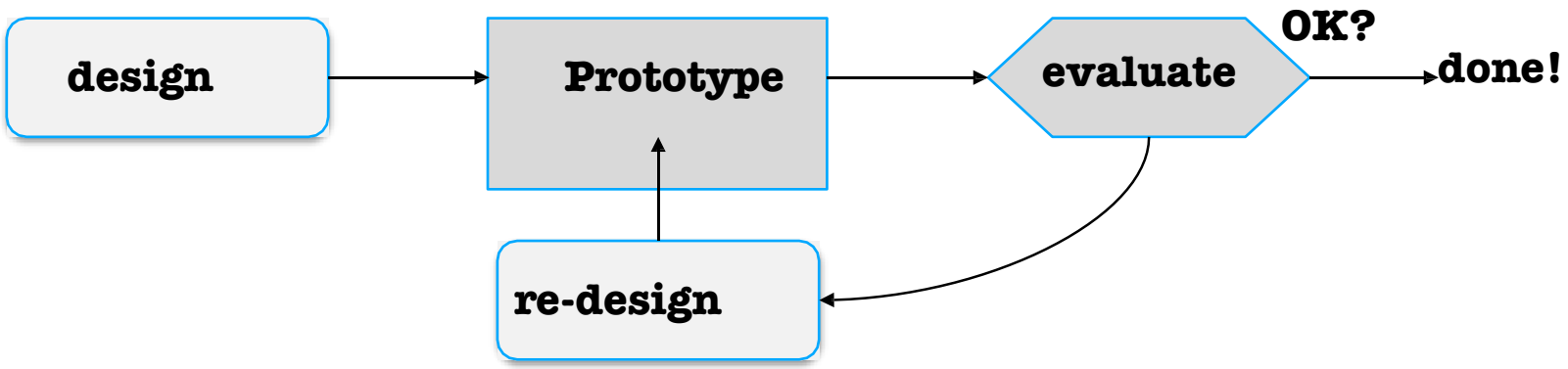
https://www.youtube.com/watch?v=JMjozqJS44M

Rapid Prototyping

- You never get it right first time
- If at first you don"t succeed ...

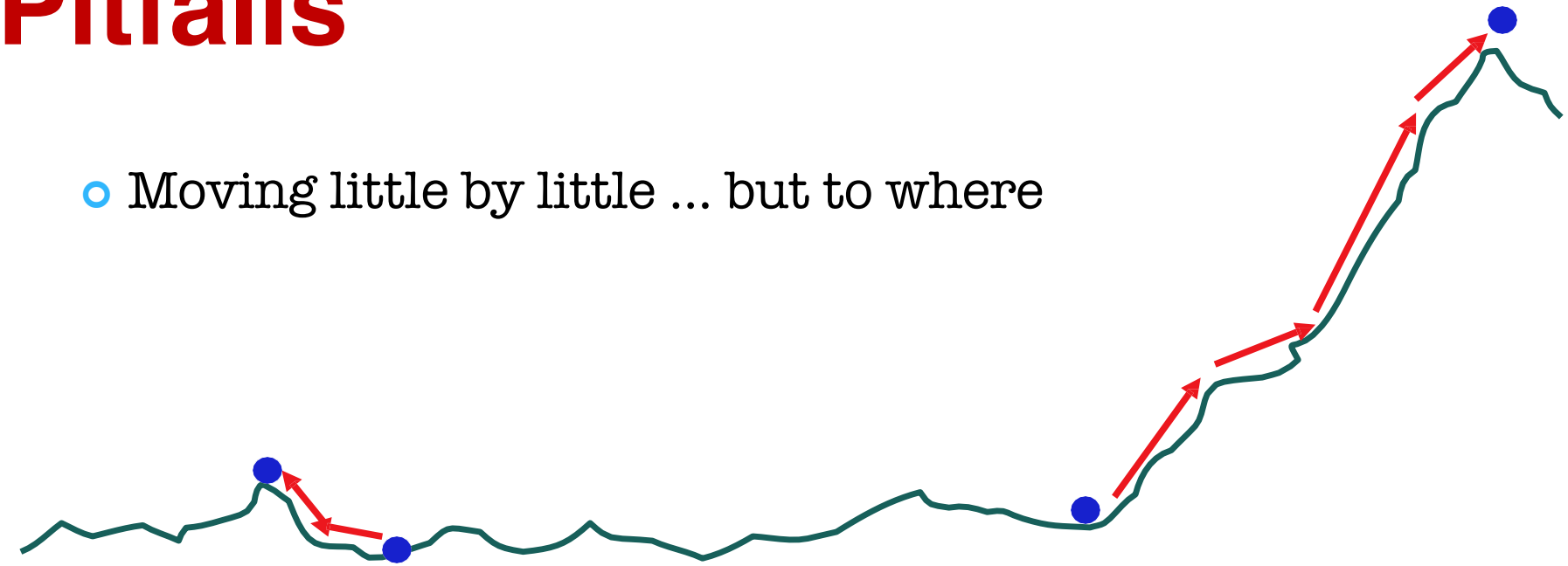


Rapid Prototyping



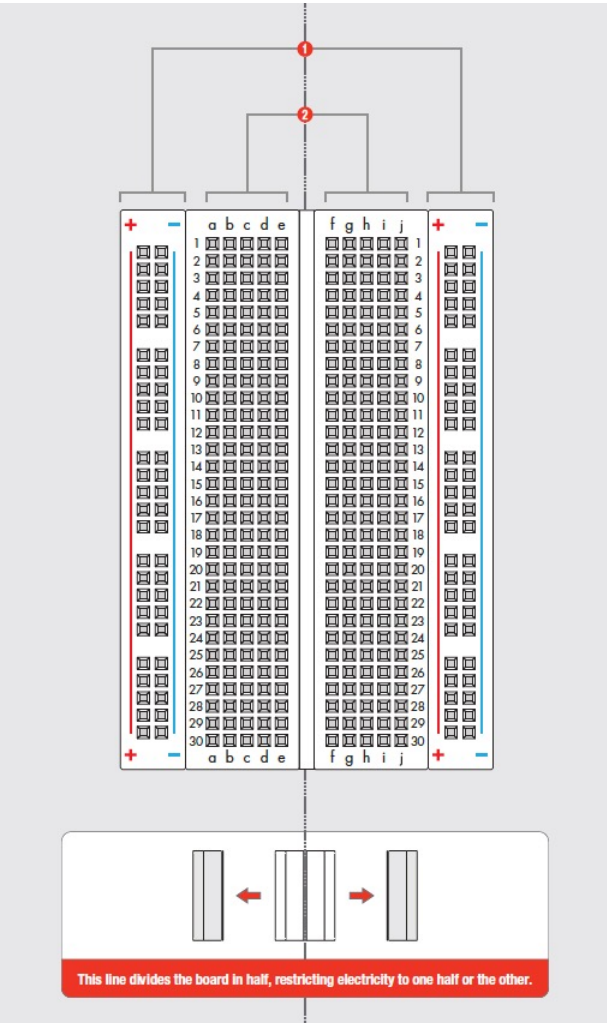
Rapid Prototyping Pitfalls

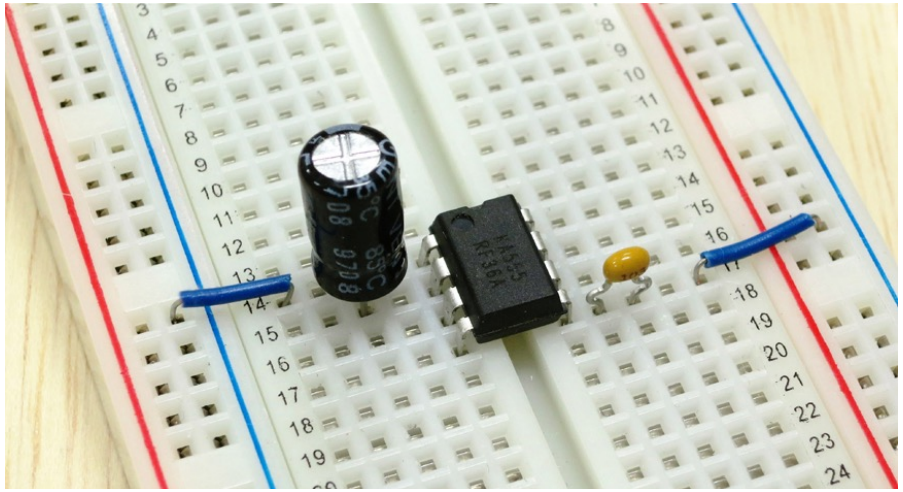
- Moving little by little … but to where

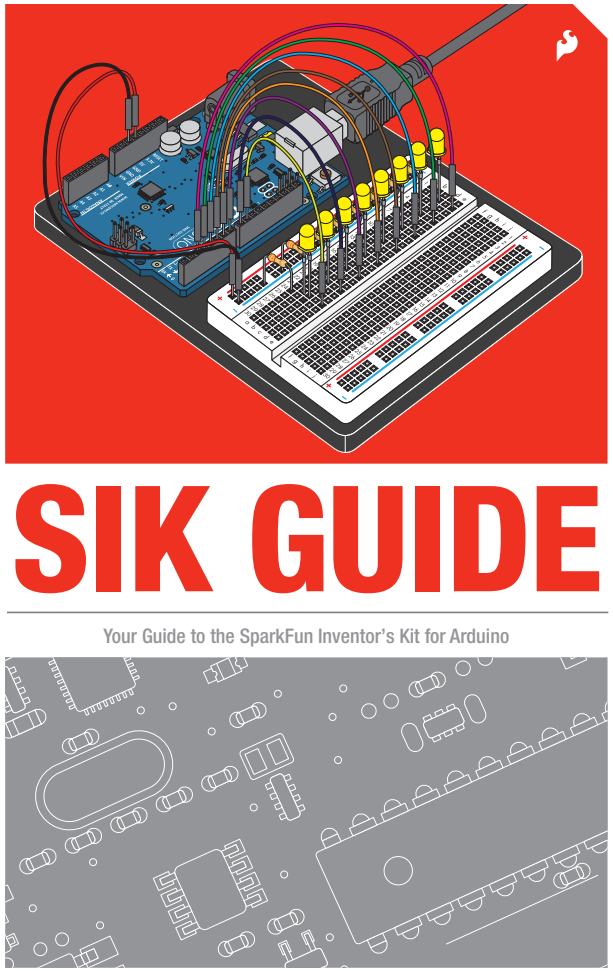


1. need a good start point
2. need to understand what is wrong

Breadboard





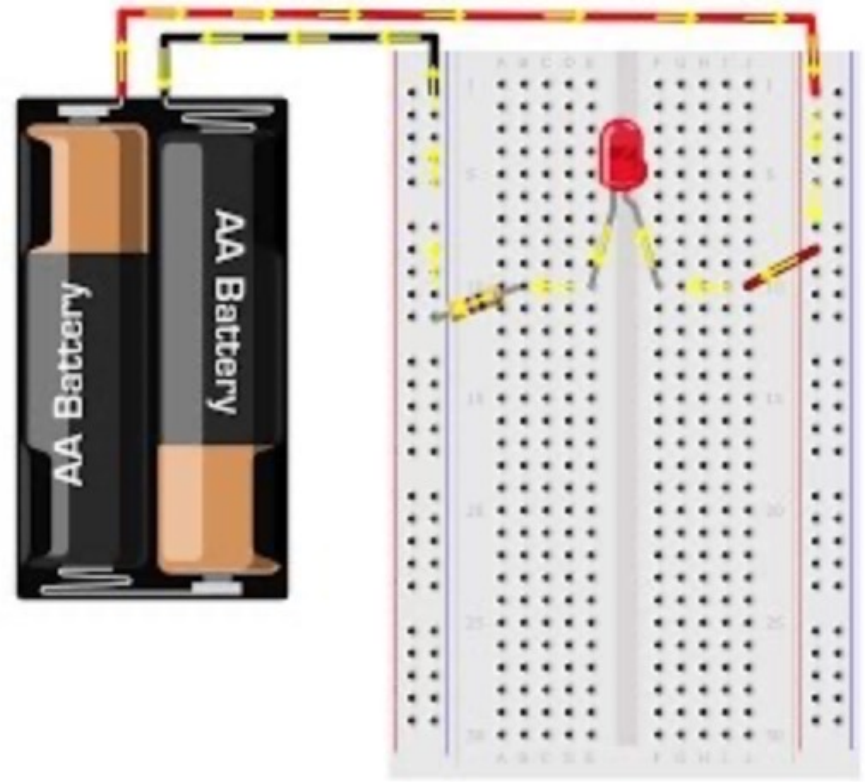


https://learn.sparkfun.com/tutorials/how-to-use-a-breadboard/

https://cdn.sparkfun.com/datasheets/Kits/RedBoard_SIK_3.2.pdf

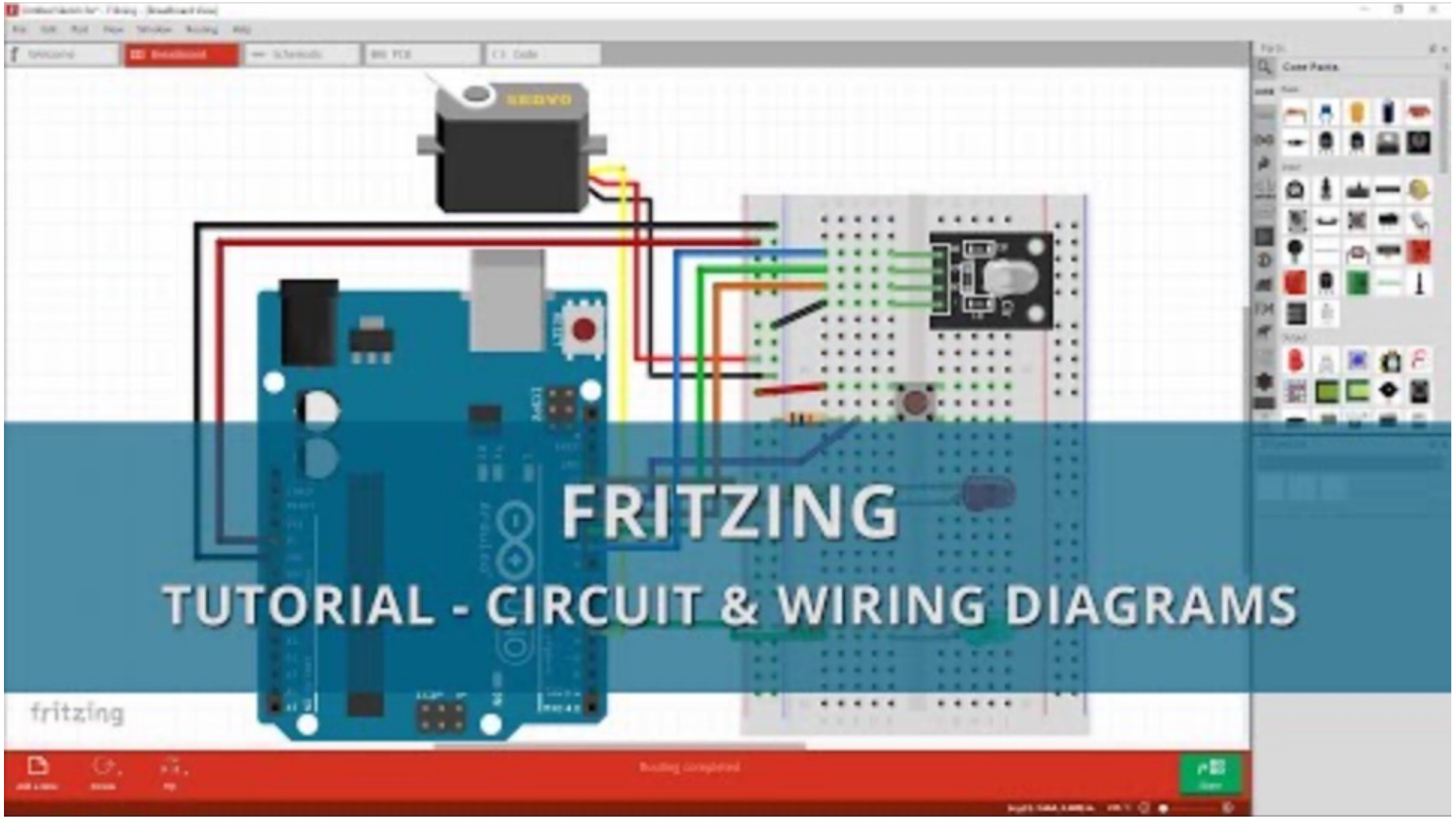
Breadboard

How to use it for Rapid Prototyping



https://www.youtube.com/watch?v=6WReFkfrUIk

Fritzing beginners guide

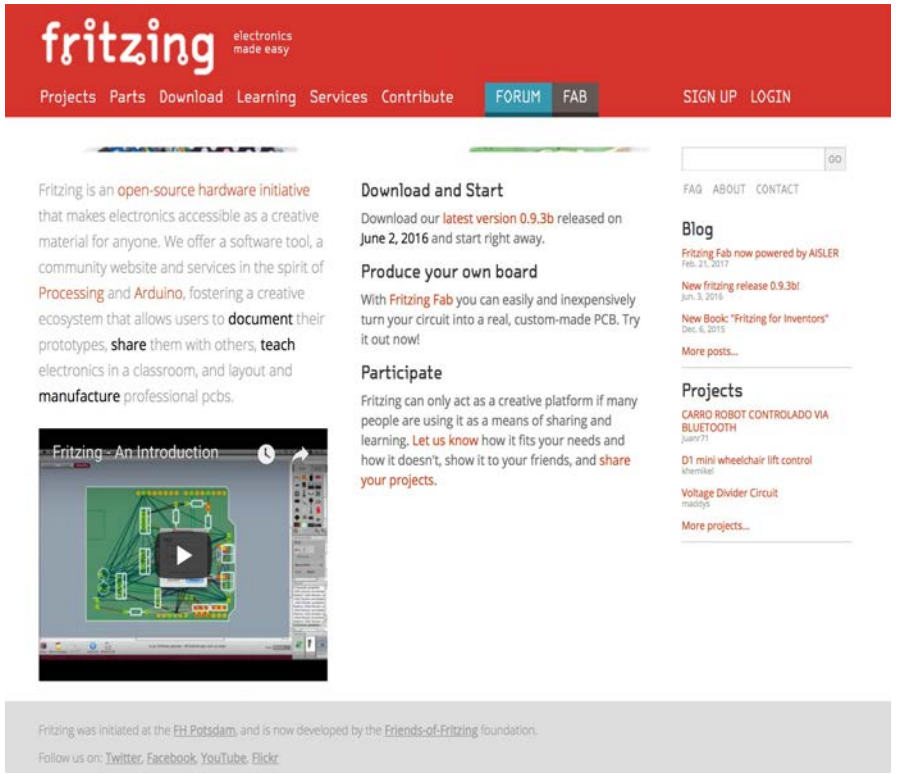


https://www.youtube.com/watch?v=-saXw1EipX0

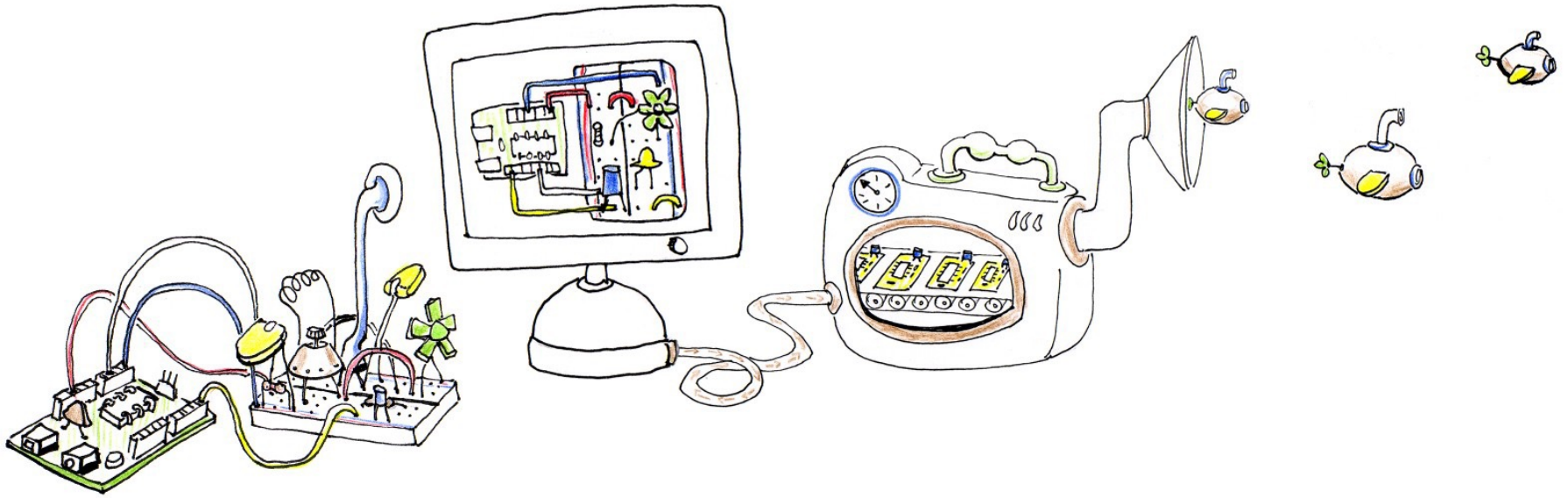
Fritzing

A tool for advancing
electronic prototyping for designers

Today a growing community of DIY-practitioners, artists and designers are using microcontroller-based toolkits to express their concepts for digital artifacts by building them.

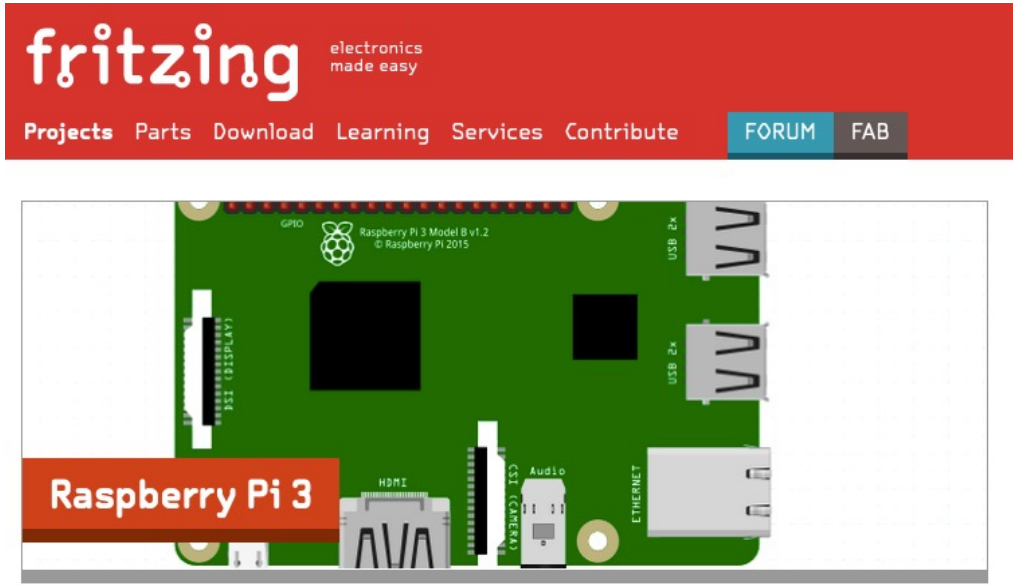


Concept of Fritzing



The recreation of their breadboard prototype in the Fritzing software enables designers to produce professional PCBs and also to share their designs.

Installing Fritzing

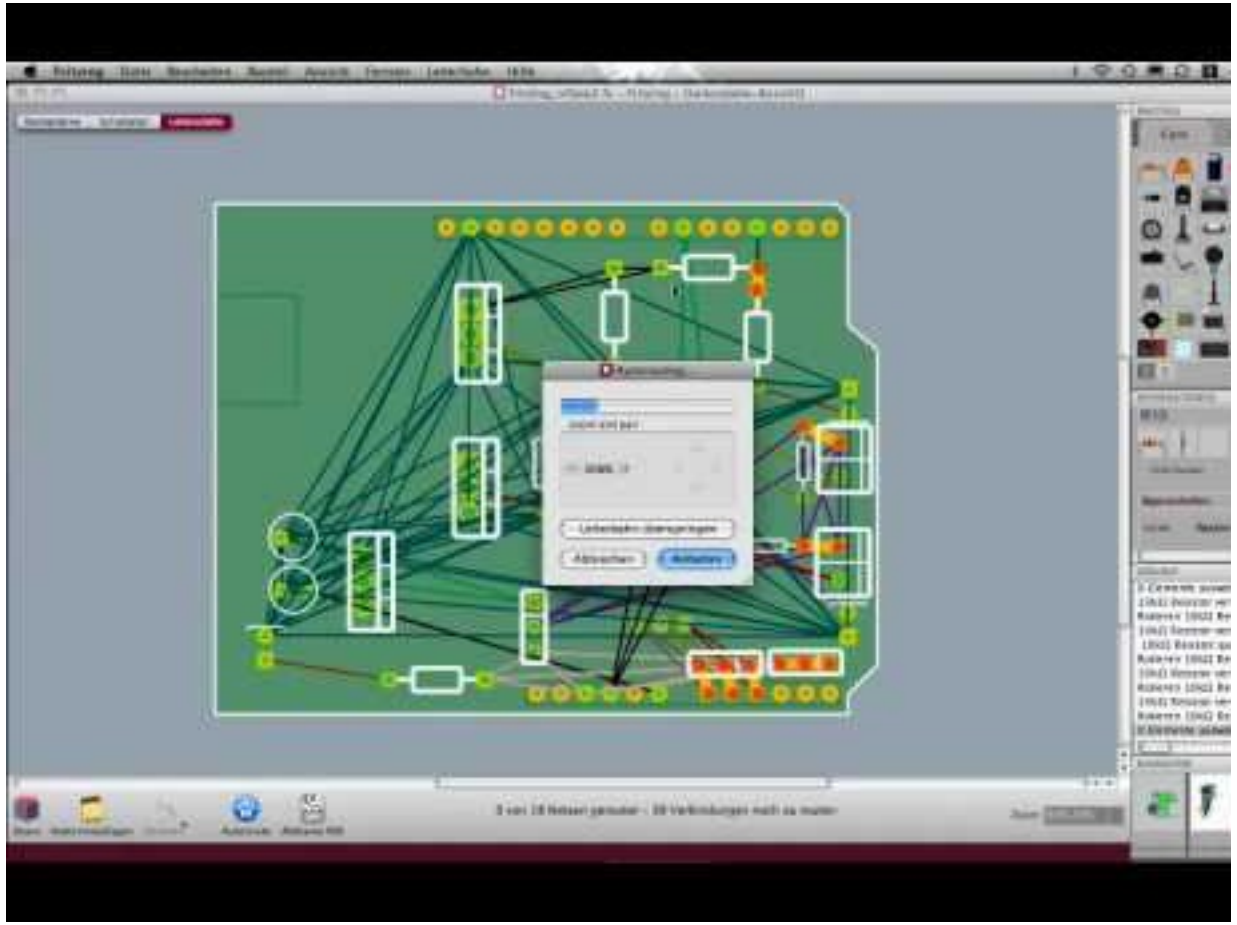


Releasing a custom fritzing part for the new Raspberrry Pi 3. Download "Raspberry_Pi3_3.fzpz".

So new Raspberry Pi 3 Model B has just been released.There are two giant upgrades in the Pi 3. The first is a next generation Quad Core Broadcom BCM2837 64-bit ARMv8 processor, making the processor speed increase from 900 MHz on the Pi 2 to up to 1.2GHz on the Pi 3.

The second giant upgrade (and this is the one we're personally most excited about) is the addition of a BCM43143 WiFi chip BUILT-IN to your Raspberry Pi. No more pesky WiFi adapters - this Pi is WiFi ready. There's also Bluetooth Low Energy (BLE) on board making the Pi an excellent IoT solution (BLE support is still in the works, software-wise)

https://fritzing.org/projects/raspberry-pi-3



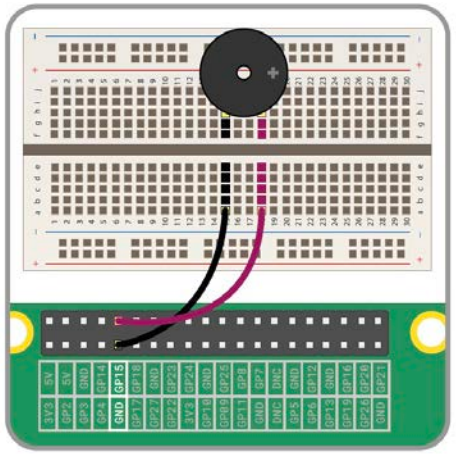
Rapid Prototyping
with Raspberry Pi

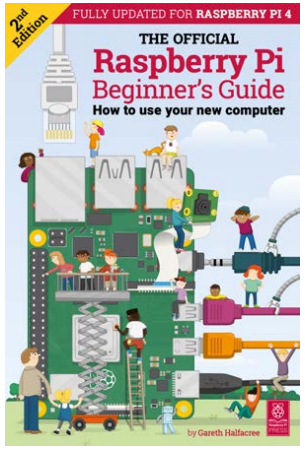
How It Works:

1 ASSEMBLE

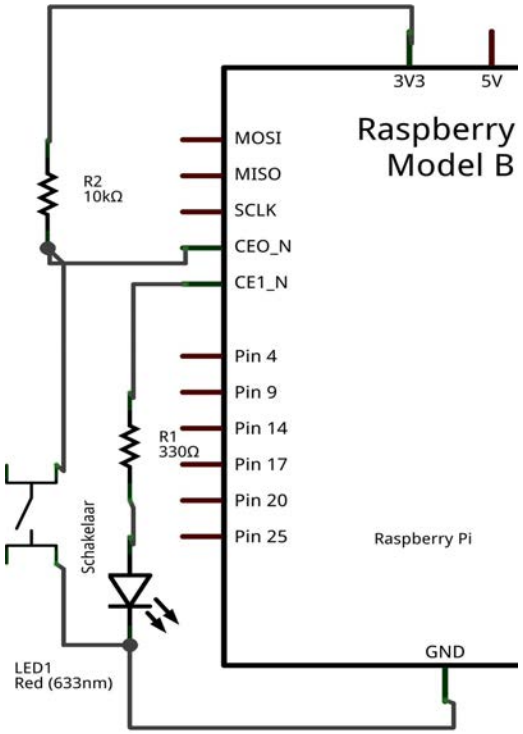
2 WRITE

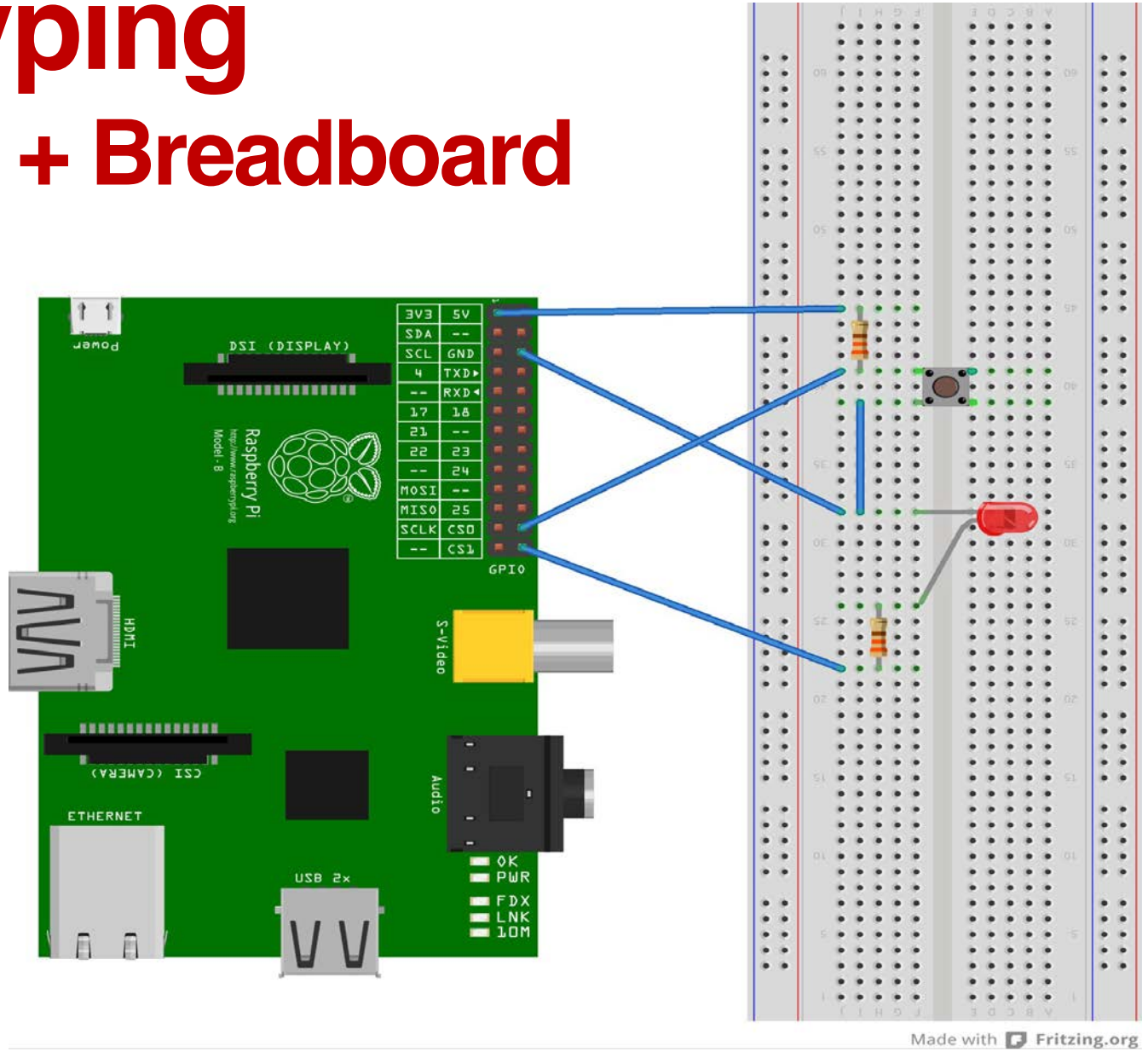
3 UPLOAD



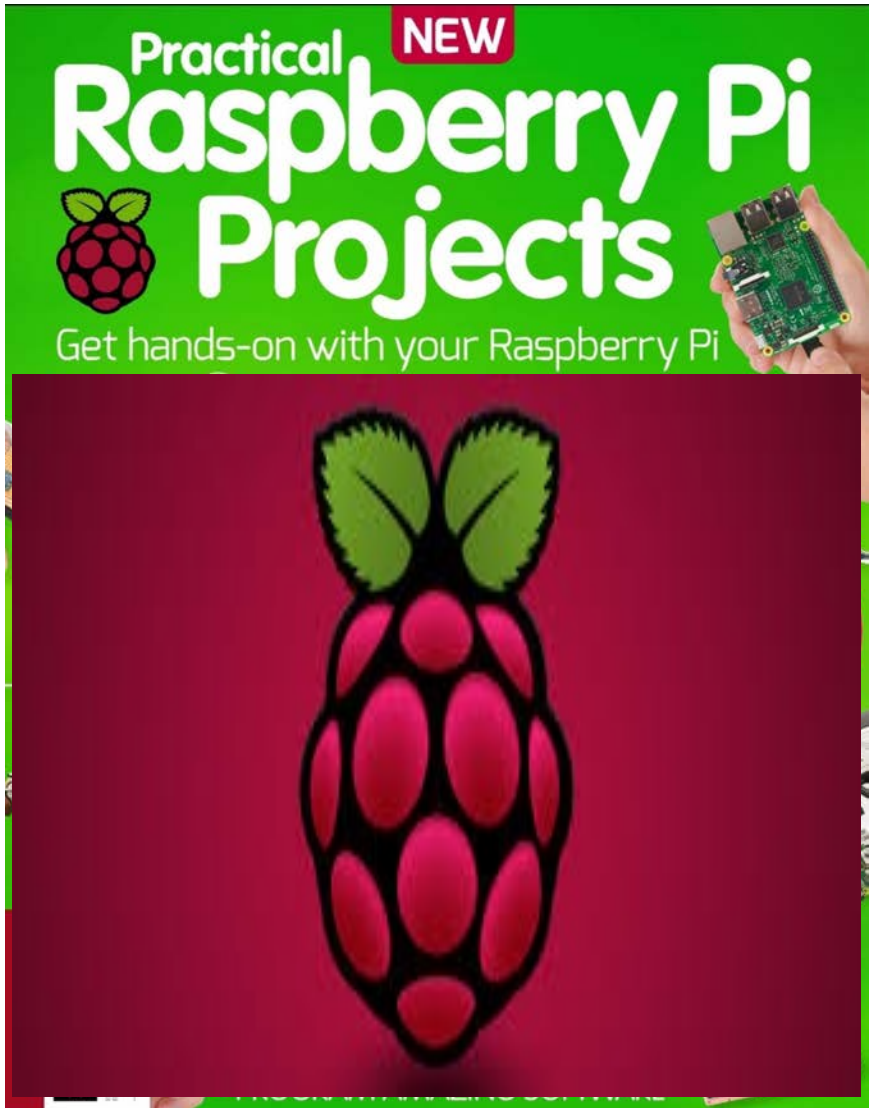


Rapid Prototyping
with Raspberry Pi + Breadboard





Raspberry Pi



Get interactive with Scratch

Experiment with physical computing by using Scratch to interact with buttons and lights on your Pi

What you'll need

- Breadboard
- LEDs
- Buttons
- Resistors
- Jumper wires
- ScratchGPIO3

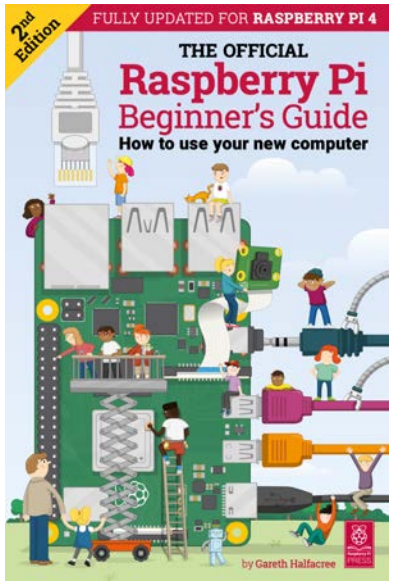
Scratch is a very simple visual programming language, commonly used to teach basic programming concepts to learners of any age. In this project we'll learn how to light up an LED when a button is pressed in Scratch, and then change a character's colour when a physical button is pressed. With these techniques you can make all manner of fun and engaging projects, from musical keyboards to controllers for your Scratch games and animations.

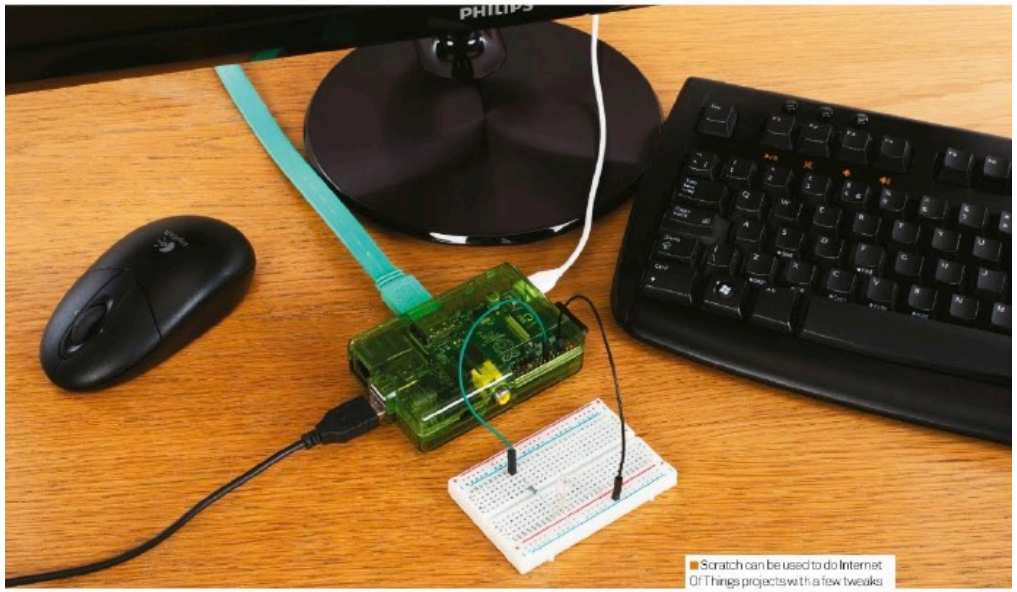
01 Installing the required software

Log into the Raspbian system with the username **Pi** and the password **raspberry**. Start the LXDE desktop environment using the command **startx**. Then open LXTerminal and type the following commands:

```
wget http://liamfraser.co.uk/lud/install_scratchgpio3.sh
chmod +x install_scratchgpio3.sh
sudo bash install_scratchgpio3.sh
```

This will create a special version of Scratch on your desktop called ScratchGPIO3. This is a normal version of Scratch with a Python script that handles communications between Scratch and the GPIO. ScratchGPIO was created by simplesi (cymplecy.wordpress.com).





Scratch can be used to do Internet Of Things projects with a few tweaks

Get interactive with Scratch

Experiment with physical computing by using Scratch to interact with buttons and lights on your Pi

What you'll need

- Breadboard
- LEDs
- Buttons
- Resistors
- Jumper wires
- ScratchGPIO3

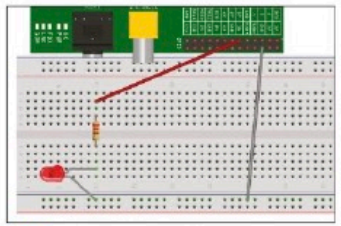
Scratch is a very simple visual programming language, commonly used to teach basic programming concepts to learners of any age. In this project we'll learn how to light up an LED when a button is pressed in Scratch, and then change a character's colour when a physical button is pressed. With these techniques you can make all manner of fun and engaging projects, from musical keyboards to controllers for your Scratch games and animations.

01 Installing the required software

Log into the Raspbian system with the username **Pi** and the password **raspberry**. Start the LXDE desktop environment using the command **startx**. Then open LXTerminal and type the following commands:

```
wget http://ziggyfraser.co.uk/lud/install_scratchgpio3.sh
chmod +x install_scratchgpio3.sh
sudo bash install_scratchgpio3.sh
```

This will create a special version of Scratch on your desktop called ScratchGPIO3. This is a normal version of Scratch with a Python script that handles communications between Scratch and the GPIO. ScratchGPIO was created by simplesi (pymplecy.wordpress.com).



02 Connecting the breadboard

Power off your Pi and disconnect the power cable. Get your breadboard, an LED, a 330-ohm resistor and two GPIO cables ready. You'll want to connect the 3.3V pin (top-right pin, closest to the SD card) to one end of the 330-ohm resistor, and then connect the positive terminal of the LED (the longer leg is positive) to the other end. The resistor is used to limit the amount of current that can flow to the LED.

Then put the negative terminal of the LED into the negative rail of the breadboard. Connect one of the GROUND pins (for example, the third pin from the right on the bottom row of pins) to the negative lane. Now connect the power to your Pi. The LED should light up. If it doesn't, then it's likely that you've got it the wrong way round, so disconnect the power, swap the legs around and then try again.

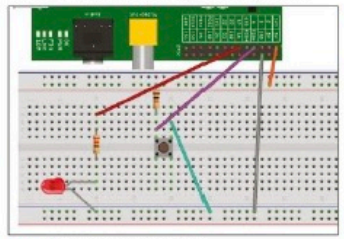
03 Switching the LED on and off

At the moment, the LED is connected to a pin that constantly provides 3.3V. This isn't very useful if we want to be able to turn it on and off, so let's connect it to GPIO 17, which we can turn on and off. GPIO 17 is the sixth pin from the right, on the top row of pins. Power the Pi back on. We can turn the LED on by exporting the GPIO pin, setting it to an output pin and then setting its value to 1. Setting the value to 0 turns the LED back off:

```
echo 17 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio17/direction
echo 1 > /sys/class/gpio/gpio17/value
echo 0 > /sys/class/gpio/gpio17/value
```

04 Controlling the LED from Scratch

Start the LXDE desktop environment and open ScratchGPIO3. Go to the control section and create a simple script that broadcasts **pin11on** when Sprite1 is clicked. Then click the sprite. The LED should light up. Then add to the script to wait 1 second and then broadcast **pin11off**. If you click the sprite again, the LED will come on for a second and then go off. ScratchGPIO3 uses pin numbers rather than GPIO numbers to identify pins. The top-right pin (the 3.3V we first connected our LED to) is pin number 1, the pin underneath that is pin number 2, and so on.



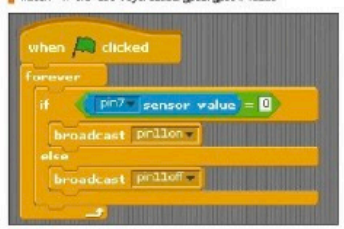
05 Wiring up our push button

Power off the Pi again. This circuit is a little bit more complicated than the LED one we created previously. The first thing we need to do is connect 3.3V (the top-right pin we used to test our LED) to the positive rail of the breadboard. Then we need to connect a 10Kohm resistor to the positive rail, and the other end to an empty track on the breadboard. Then on the same track, add a wire that has one end connected to GPIO 4. This is two pins to the right of GPIO 17. Then, on the same track again, connect one pin of the push button. Finally, connect the other pin of the push button to ground by adding a wire that is connected to the same negative rails that ground is connected to.

When the button is not pressed, GPIO 4 will be receiving 3.3V. However, when the button is pressed, the circuit to ground will be completed and GPIO 4 will be receiving 0V (and have a value of 0), because there is much less resistance on the path to ground.

We can see this in action by watching the pin's value and then pressing the button to make it change:

```
echo 4 > /sys/class/gpio/export
echo in > /sys/class/gpio/gpio4/direction
watch -n 0.5 cat /sys/class/gpio/gpio4/value
```



06 Let there be light!

Boot up the Pi and start ScratchGPIO3 as before. Go to the control section and add **when green flag clicked**, then attach a **forever** loop, and inside that an if **else** statement. Go to the operators section and add an if [] = [] operator to the if statement. Then go to the sensing section and add a value sensor to the left side of the equality statement, and set the value to pin7. On the right side of the equality statement, enter 0. Broadcast **pin11on** if the sensor value is 0, and broadcast **pin11off** otherwise. Click the green flag. If you push the button, the LED will light up!

You'll learn...

1. Simple circuits
While these are very simple circuits, you'll get a great feel of how the Raspberry Pi interfaces with basic prototyping kit. If you need to buy the bits and pieces, we recommend you check out: shop.pimoroni.com

2. Coding principles
If you're new to programming, Scratch is the perfect place to learn the same programming principles employed by all programming languages.

3. Physical computing
There's nothing more magical than taking code from your computer screen and turning it into a real-life effect. Your first project might just turn a light on and off, but with that skill banked, the sky is the limit.

Things you should Do

HOGESCHOOL ROTTERDAM

To Do's

DIY Blinking an LED on the Raspberry Pi

https://embeddedcode.wordpress.com/2017/01/18/blinking-an-led-on-the-raspberry-pi/

Do it yourself (DIY) : Make a LED blinking

Raspberry Pi, Power Supply,

microSD Card with Raspbian Installed and Setup,

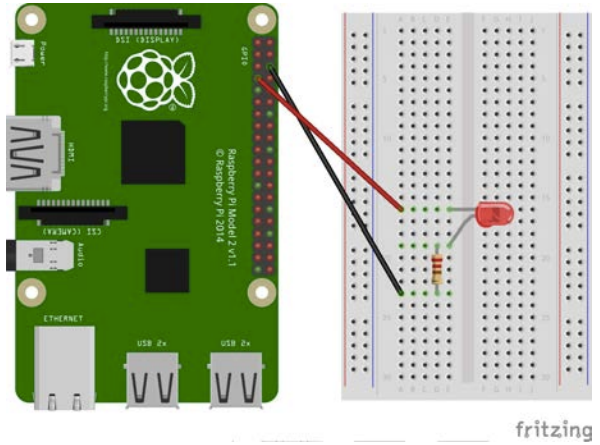
Internet connectivity (Optional),

Bread board,

connecting wires,

an LED 5mm of preferred colour

and a resistor of 220Ohms



LEARNING OBJECTIVES

In this Rapid prototyping experiment,
You will learn the essentials of Raspberry Pi GPIO control by toggling an LED at predefined intervals of time.

KEY WORDS, CONCEPTS, & PRACTICES

+ LED
+ Resistor
+ GPIO
+ Rapid Prototyping
+ Coding (Python)

Reminder lesson 01

1st STEP Managing Data Science for IoT Projects

➔➔➔Getting started with GitHub➔➔➔

A Computational Thinking culture has an intellectual dimension, engaging with a set of creative concepts and practices. It has a physical dimension, encouraging interactions with others through the placement of desks, chairs, and computers. Most importantly, it has an affective dimension, cultivating a sense of confidence and fearlessness.

LEARNING OBJECTIVES

Students will:

+ be introduced to the concept of computational Thinking, in the context of GitHub
+ be able to imagine possibilities for their own GitHub-based computational thinking
+ become familiar with resources that support their computational thinking
+ prepare for creating GitHub projects by establishing a GitHub account, exploring GitHub, creating design journals

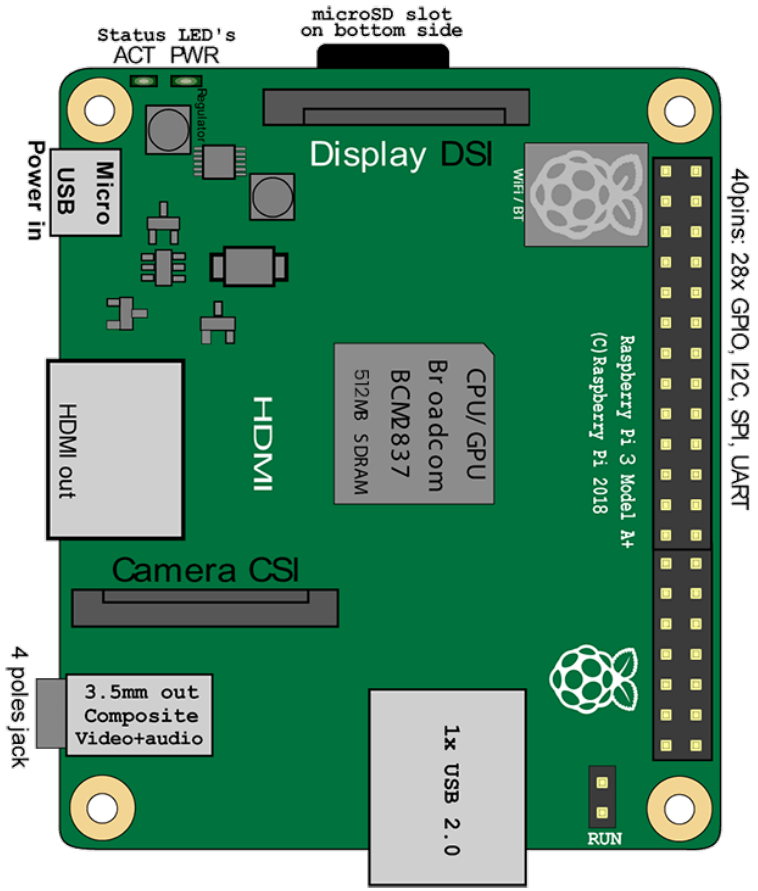
KEY WORDS, CONCEPTS, & PRACTICES

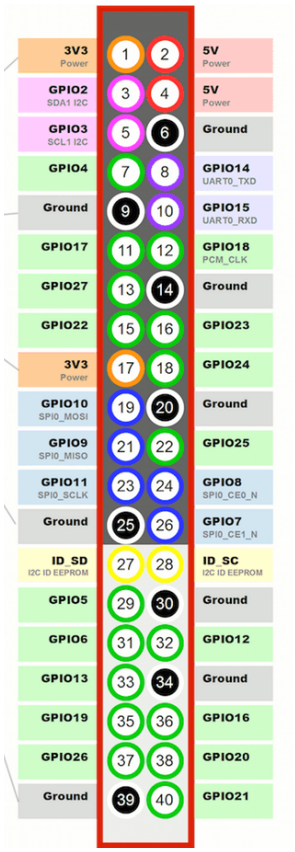
+ GitHub
+ Computational Thinking

Preview
lesson Three

HOGESCHOOL ROTTERDAM

Raspberry Py 3A+



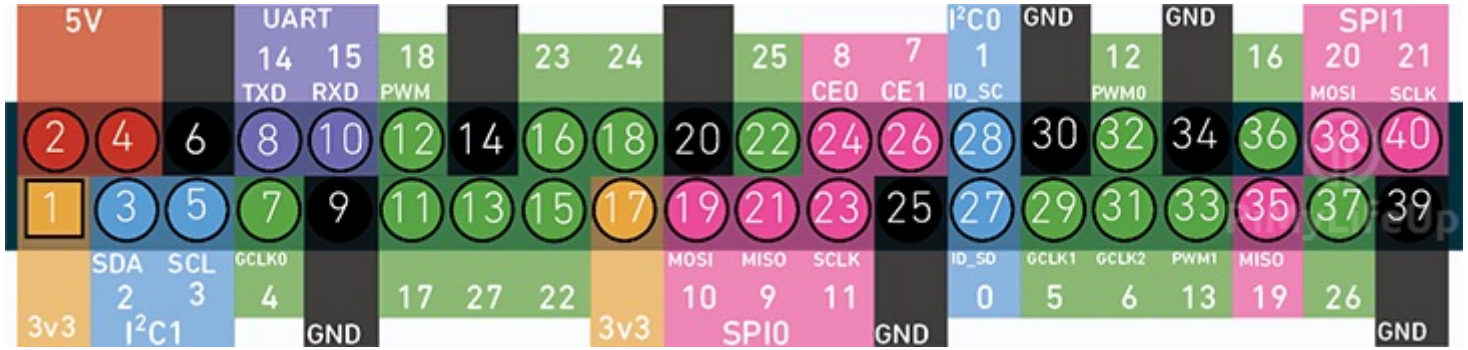


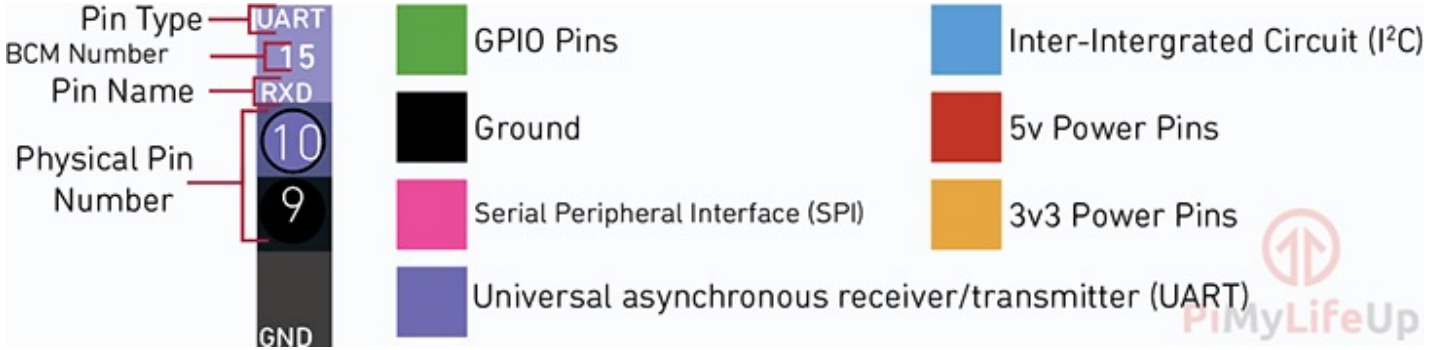
Here we have included some schematics of the Raspberry Pi 3 A Plus hardware.
These schematics show the general positioning of all the vital circuitry on the board.
The Raspberry Pi 3 A+ is a cut down version of the Pi 3B. As you can tell by its diagram, it features a single USB 2.0 Port. It's only means of network connectivity is the inbuilt Wi-Fi.

CPU: 1.4 GHz quad core ARM Cortex-A53
GPU: 250MHz Broadcom VideoCore IV
RAM: 512mb (Shared with GPU)
Storage: Micro SD
USB 2.0 Ports: 1
USB 3.0 Ports: 0
Networking: 802.11b/g/n/ac dual band 2.4/5 GHz wireless, Bluetooth 4.2 LS BLE
Video Input: 15-pin MIPI camera interface (CSI) connector
Video Ouputs: HDMI 1.3, MIPI display interface, DSI
Audio Inputs: Audio over I2S
Audio Outputs: 3.5mm phone jack, Digital Audio via HDMI
Low-Level peripherals: 17 x GPIO, +3.3v, +5v, ground, Plus the following that can be used as GPIO: UART, I2C Bus, SPI bus with two chip select, I2S audio
Power Source: 5v via MicroUSB or GPIO header
Size: 65.00mm x 56.50mm x 17mm
Weight: 23 g (0.81 oz)

GPIO (general-purpose input/output)

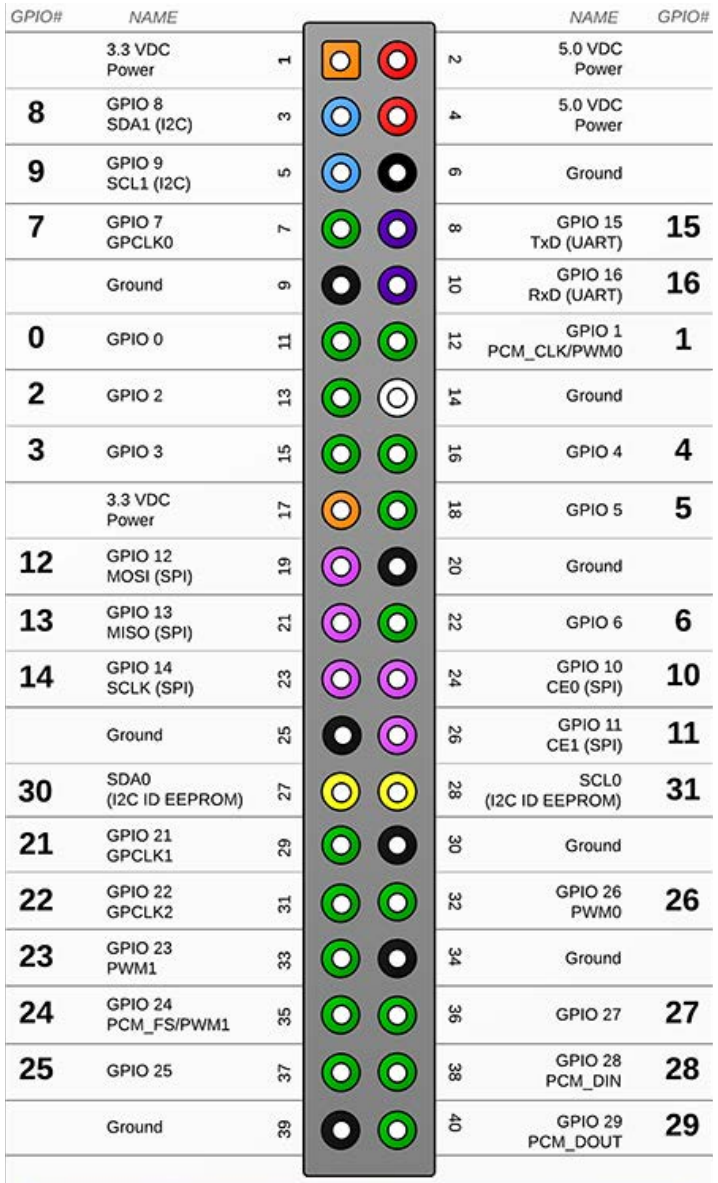
connecting to the outside world





https://pimylifeup.com/raspberry-pi-3-a-plus/

GPIO pins groups on the Raspberry 3



GPIO#	NAME				NAME	GPIO#
	3.3 VDC Power	1	2		5.0 VDC Power	
8	GPIO 8 SDA1 (I2C)	3	4		5.0 VDC Power	
9	GPIO 9 SCL1 (I2C)	5	6		Ground	
7	GPIO 7 GPCLK0	7	8		GPIO 15 TxD (UART)	15
	Ground	9	10		GPIO 16 RxD (UART)	16
0	GPIO 0	11	12		GPIO 1 PCM_CLK/PWM0	1
2	GPIO 2	13	14		Ground	
3	GPIO 3	15	16		GPIO 4	4
	3.3 VDC Power	17	18		GPIO 5	5
12	GPIO 12 MOSI (SPI)	19	20		Ground	
13	GPIO 13 MISO (SPI)	21	22		GPIO 6	6
14	GPIO 14 SCLK (SPI)	23	24		GPIO 10 CE0 (SPI)	10
	Ground	25	26		GPIO 11 CE1 (SPI)	11
30	SDA0 (I2C ID EEPROM)	27	28		SCL0 (I2C ID EEPROM)	31
21	GPIO 21 GPCLK1	29	30		Ground	
22	GPIO 22 GPCLK2	31	32		GPIO 26 PWM0	26
23	GPIO 23 PWM1	33	34		Ground	
24	GPIO 24 PCM_FS/PWM1	35	36		GPIO 27	27
25	GPIO 25	37	38		GPIO 28 PCM_DIN	28
	Ground	39	40		GPIO 29 PCM_DOUT	29

https://pi4j.com/1.1/pins/model-3b-rev1.html

•**Power:** Pins that are labeled 5.0v supply 5 volts of power and those labeled 3V3 supply 3.3 volts of power. There are two 5V pins and two 3V3 pins.
•**GND:** These are the ground pins. There are eight ground pins.

•**Input/Outputpins**: These are the pins labeled with the # sign, for example, #17, #27, #22, etc. These pins can be used for input or output.

•**I2C:** I2C is a serial protocol for a two-wire interface to connect low-speed devices like microcontrollers, EEPROMs, A/D and D/A converters, I/O interfaces, and other similar peripherals in embedded systems. These pins are labeled **SDA** and **SCL**.

•**UART:** The **Universal Asynchronous Receiver/Transmitter** allows your Raspberry Pi to be connected to serial peripherals. The UART pins are labeled **TXD**and **RXD**.

•**SPI:** The **Serial Peripheral Interface** is a synchronous serial communication interface specification used for short distance communication, primarily in embedded systems. The SPI pins are labeled **MOSI**, **MISO**, **SCLK**, **CE0**, and **CE1**.

•**ID EEPROM:Electrically Erasable Programmable Read-Only Memory** is a user-modifiable read-only memory that can be erased and written to repeatedly through the application of higher than normal electrical voltage. The two EEPROM pins on the Raspberry Pi (**EED** and **EEC**) are also secondary I2C ports that primarily facilitate the identification of Pi Plates (e.g., Raspberry Pi Shields/Add-On Boards) that are directly attached to the Raspberry Pi.

GPIO general purpose IO

Voorgedefinieerde pennen

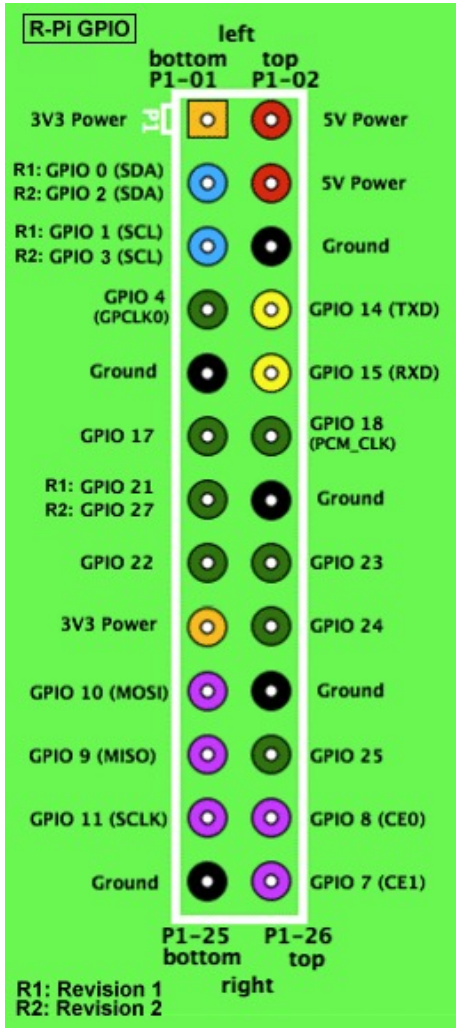
- 0V, 3.3V, 5V, Transmit, Receive
- I2C, 1-Wire

Vrijbeschikbare pennen

- GPIO 4, 17, 18, 8, 7

Naamgeving

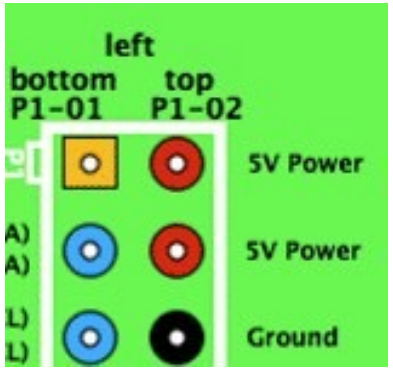
- Pinnummer van connector P1
- Broadcom documentatie van de SoC (System on Chip)



GPIO

Voordat je iets aansluit

- input maximaal 3.3V
  - let met name op de 5V van pin 2 en 4
- output maximaal 16 mA
  - LED over 330 Ω weerstand mag
  - motortje heeft een buffer nodig (ULN2003)
- totale output maximaal 50 mA



Programmeren

Schrijven (led laten knipperen)

- Eerst: GPIO pennetje definieren als output
- Dan: herhaaldelijk schrijven

Pennetje 26 = GPIO 7

Lezen (schakelaar uitlezen)

- Eerst: GPIO pennetje definieren als input
- Dan: herhaaldelijk lezen

Pennetje 24 = GPIO 8

Programmeren (bash shell)

- bash is de Linux command line interpreter
- als root in directory /sys/class/gpio werken
- GPIO 7 voor uitvoer

```
echo "7" >/sys/class/gpio/export
echo "out" >/sys/class/gpio/gpio7/direction
```

- led aan (1), led uit (0)

```
echo   "1"      > /sys/class/gpio/gpio7/value
echo   "0"      > /sys/class/gpio/gpio7/value
```

Programmeren (bash shell)

Knipperen

```
while sleep 0.5
do echo "1" > /sys/class/gpio/gpio7/value  sleep  0.5
   echo "0" > /sys/class/gpio/gpio7/value  done
```

GPIO 8 voor invoer

```
echo "8" > /sys/class/gpio/export
echo "in" > /sys/class/gpio/gpio8/direction
```

eenmalig lezen

```
cat /sys/class/gpio/gpio8/value
```

Programmeren (bash shell)

herhaald lezen

```
while sleep 0.1
do cat /sys/class/gpio/gpio8/value
done
```

opruimen

```
echo "7"      > /sys/class/gpio/unexport
echo "8"      > /sys/class/gpio/unexport
```

Programmeren (python)

Pi in Raspberry Pi staat voor Python

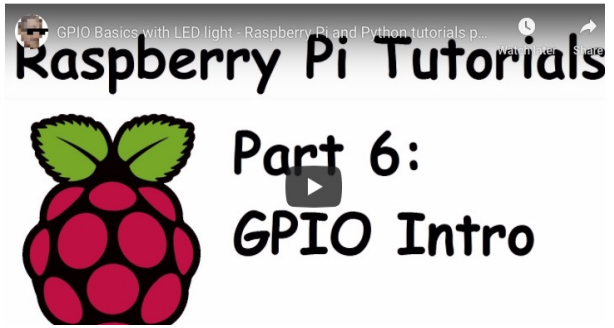
Standaard modules en bibliotheken

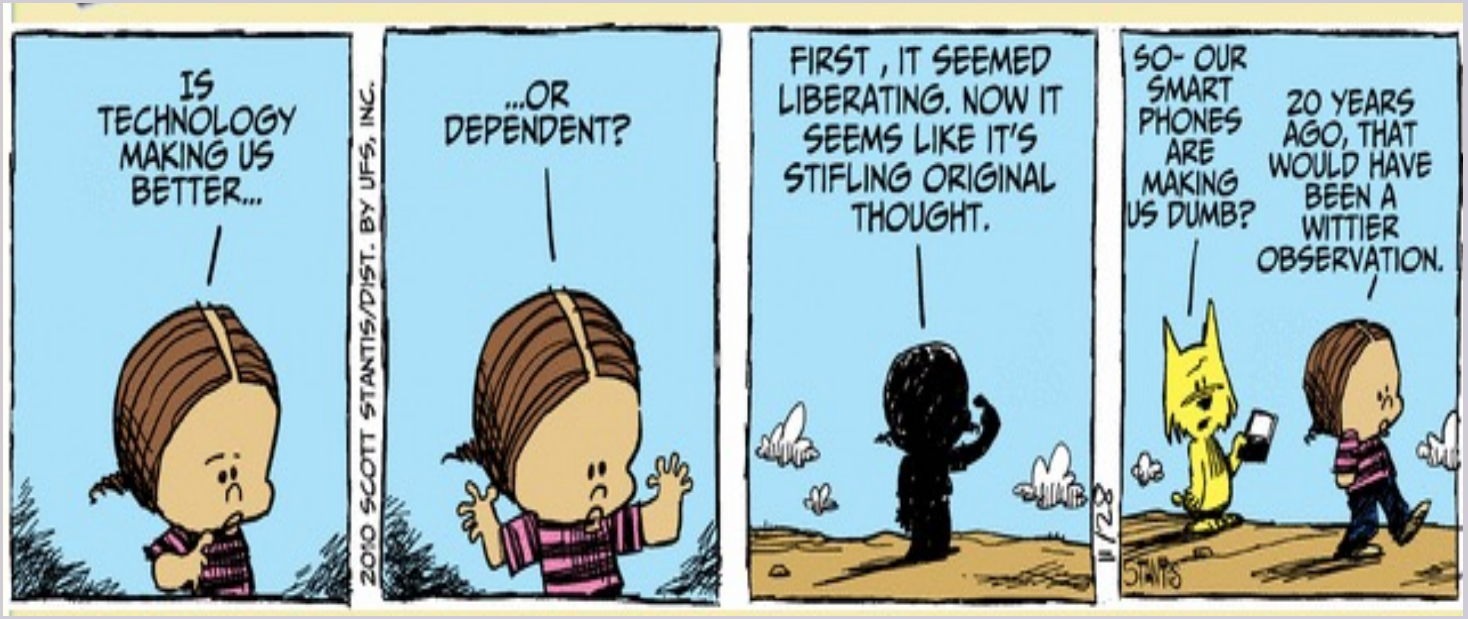
- keuze in naamgeving pennetjes / meer functies
- https://pythonprogramming.net/gpio-raspberry-pi-tutorials/

RPi.GPIO

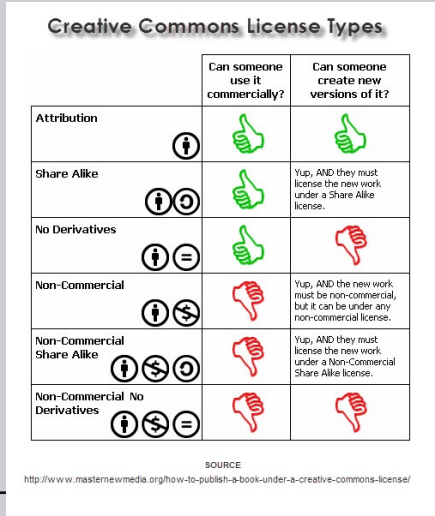
- standaard onderdeel van Raspbian
- https://pypi.org/project/RPi.GPI

GPIO (General Purpose Input Output) Pins - Raspberry Pi tutorial





This seminar was developed by:
Rob van der Willigen
LivingLab AI & Ethics
Hogeschool Rotterdam
November 2021



HOGESCHOOL ROTTERDAM