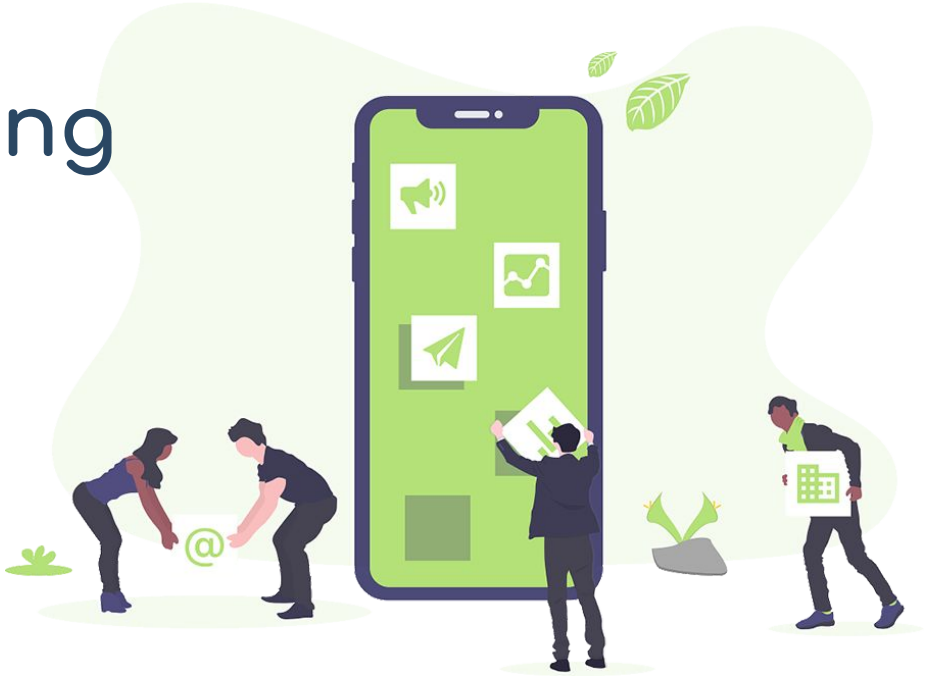
HELLO WORLD

# Digital Skills & Coding

## Coding for Non-Coders Beiersdorf

25.09.2019



# Agenda

1. Setup & Tools
2. Python 101
3. Excel is dead, long live Pandas!
4. Automate the boring stuff
5. User Journey of a Website
6. Introduction to html, CSS & JS
7. Webscrapping with Python
8. You did it!

# Your trainers

Dr. rer. nat.
Joachim Krois

**Areas of interest**
Coding, Predictive Modelling, Spatial Data Analytics, Artificial Intelligence and Computer Vision

**References**
Freie Universität Berlin, Charité – Universitätsmedizin Berlin, Berliner Institut für Gesundheits-forschung

Annemieke
Frank

**Areas of interest**
Co-Learning & Digitale Kompetenzen, Coding, Design Thinking, Gamification

**References**
Facebook, Volkswagen, Deutsche Bahn, Code+Design, Technologie Stiftung

# Let's connect

https://etherpad.hello-world.academy/p/beiersdorf

# Let's have some fun :)

Visit the following website on your phone and type in the GAME PIN.

## www.kahoot.it

The questions and answers will appear on the main screen upfront and you choose your answers on your phone.

Speed counts too 😎

# How to Create a Website

## An easy, step-by-step guide for beginners

# How to Create a Website

## An easy, step-by-step guide for beginners

### STEP 1: Get a domain

What is a domain?

# How to Create a Website

## An easy, step-by-step guide for beginners

### STEP 1: Get a domain

What is a domain?

- A domain name is an identification string
- Domain names are easy-to-remember words that we can use to tell a Domain Name System (DNS) server the website we want to visit.
- The DNS is what translates the friendly name to an IP address.
- Domain names are organized right to left, with general descriptors to the right, and specific descriptors to the left.

# How to Create a Website

## An easy, step-by-step guide for beginners

### STEP 1: Get a domain

What is a domain?

Second-Level Domain (SLD)

www.hello-world.academy

Top-Level Domain (TLD)

# How to Create a Website

## An easy, step-by-step guide for beginners

### STEP 1: Get a domain

Services:

- www.hetzner.de
- www.1und1.de
- www.strato.de
- www.godaddy.com
- www.domainpreisvergleich.de

# How to Create a Website

## An easy, step-by-step guide for beginners

### STEP 2: Choose a Webhoster

What is a webhoster?

# How to Create a Website

## An easy, step-by-step guide for beginners

### STEP 2: Choose a Webhoster

What is a webhoster?

- A webhoster is a home for your website.  This is just the same as choosing and registering your business and after that choosing a place for your office or shop and then renting it.
- Technically, a web host is a company which has numerous computers connected to internet. Only after placing your web pages on a web host, you let people to access them, connect to them and view them.

# How to Create a Website

## An easy, step-by-step guide for beginners

### STEP 2: Choose a Webhoster

How to find a good Webhoster?
There is only one choice: FREE vs. COMMERCIAL

# How to Create a Website

## An easy, step-by-step guide for beginners

### STEP 2: Choose a Webhoster

How to find a good Webhoster?
There is only one choice: FREE vs. COMMERCIAL

- Advertising
- Amount of web space
- FTC access
- File type & size
- Reiability & speed
- PHP
- Bandwidth
- Technical Support
- SSL
- Email, Autoresponders, POP3, Mail Forwarding
- Control Panel
- Multiple Domain Hosting & Subdomains
- International

# How to Create a Website

## An easy, step-by-step guide for beginners

### STEP 3: Design a Website

- You can design your web pages yourself or hire a designer to do it for, after you have set your domain name and web host.

# How to Create a Website

## An easy, step-by-step guide for beginners

### STEP 3: Design a Website

- You can design your web pages yourself or hire a designer to do it for, after you have set your domain name and web host.
- There are plenty of ways of how to build a website:

Learn to Code:

HTML

CSS

PHP

...

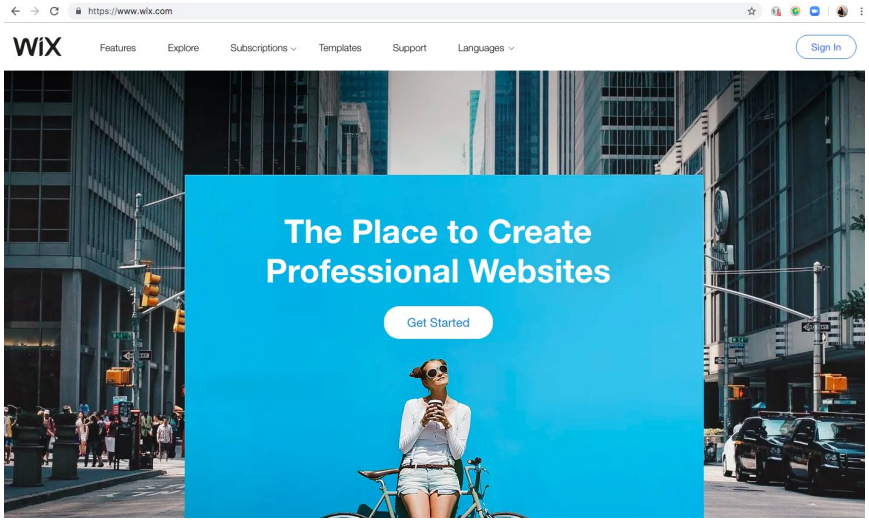
Use a website builder

WordPress, Wix, Joomla, Progress Sitefinity, SquareSpace, Weebly

# How to Create a Website

## An easy, step-by-step guide for beginners

### STEP 3: Design a Website



# How to Create a Website

## An easy, step-by-step guide for beginners

### STEP 4: Getting your site noticed

How do you get your site noticed?

# How to Create a Website

## An easy, step-by-step guide for beginners

### STEP 4: Getting your site noticed

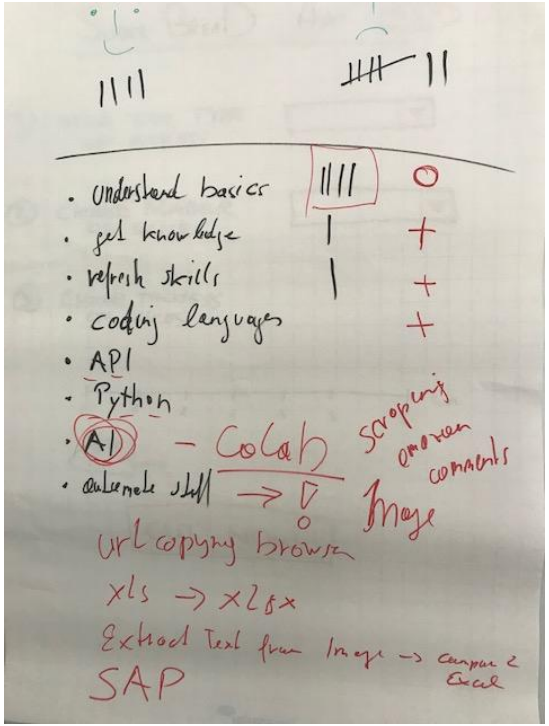
How do you get your site noticed?

- You may want to submit it to prominent search engines: being Bing and Google: Google's Link Submission Page
- SEO: It's the practice of optimizing your web pages to make them reach a high position in the search results of Google or other search engines.

# Wrap Up

- Did we meet your expectations?
- What went particularly well?
- What shall be improved?

# Expectations





Annemieke Frank
Dr. Joachim Krois

www.hello-world.academy
hello@hello-world.academy

# A brief introduction into AI

## Morgan Film Trailer

- The trailer for the 2016 film Morgan - a film about a rogue artificial intelligence - was created by IBM's AI Watson.
- Watson studied more than 100 horror movie trailers to identify the perfect trailer, which it then built.
- Watson built the trailer in 24 hours, rather than 6 weeks that it would take for a human.

# A brief introduction into AI

## Cleverbot

- A chatterbot web application that uses an artificial intelligence (AI) algorithm to have conversations with humans.
- It was created by British AI scientist Rollo Carpenter.
- It was preceded by Jabberwacky, a chatbot project that began in 1986 and went online in 1997.[

# A brief introduction into AI

## What is AI?

- The concept of what defines AI has changed over time, at the core: Building machines which are capable of thinking like humans.
- AI, can be thought of as:

  Simulating the capacity for abstract, creative, deductive thought – and particularly the ability to learn – using the digital, binary logic of computers.

# A brief introduction into AI

**NARROW AI**

Simulating human thought to carry out one specific task.

- Quantum physics
- Medicin
- Financial World
- Manufacturing
- Siri & Google Assist
- Self-driving Cars

**GENARALIZED AI**

Seeks to develop machine intelligences that can do any task, much like a person.

Generalized AI is a bit further off – to carry out a complete simulation of the human brain would require both a more complete understanding of the organ than we currently have, and more computing power than is commonly available to researchers.

# A brief introduction into AI

## IBM Watson Super Computer

- Watson is an IBM supercomputer that combines AI and sophisticated analytical software for optimal performance as a "question answering" machine.
- Watson accesses 90 servers with a combined data store of over 200 million pages of information, which it processes against six million logic rules. The system and its data are self-contained in a space that could accommodate 10 refrigerators.
- Healthcare was one of the first industries to recommend treatment options for lung cancer patients to ensure they received the right treatment while reducing costs

# A brief introduction into AI

## IBM Watson Super Computer

- Watson is an IBM supercomputer that combines AI and sophisticated analytical software for optimal performance as a "question answering" machine.
- Watson accesses 90 servers with a combined data store of over 200 million pages of information, which it processes against six million logic rules. The system and its data are self-contained in a space that could accommodate 10 refrigerators.
- Healthcare was one of the first industries to recommend treatment options for lung cancer patients to ensure they received the right treatment while reducing costs

# A brief introduction into AI

## Google Brain

- Inside it's high-tech R&D "X" laboratory the search giant, Google has been creating a simulation of the human brain.
- And rather than teaching it programs, Google's staff have been exposing it to information from the Net so that it learns organically, a little like the way we humans do.

# The Bread Machine

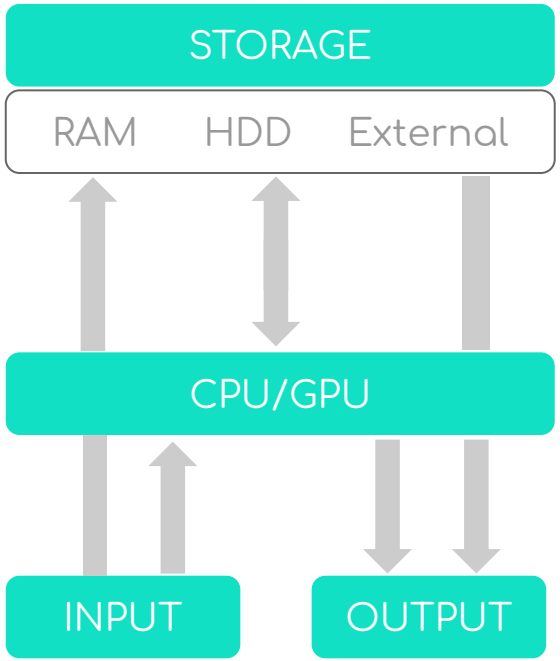
1. Give the robot commands so that 3 same-sized pieces of baguette are cut.
2. Write down each command on a post-it.
3. Sort post-its in the right order.
4. Create a mockup with (a) the type of bread, (b) the number of pieces & (c) the thickness of the slices.

# Agenda

1. <Hello World>
2. Computational Thinking
3. Theory & Terminology
4. Programming Robots
5. Coding & Languages
6. App Prototyping
7. You did it

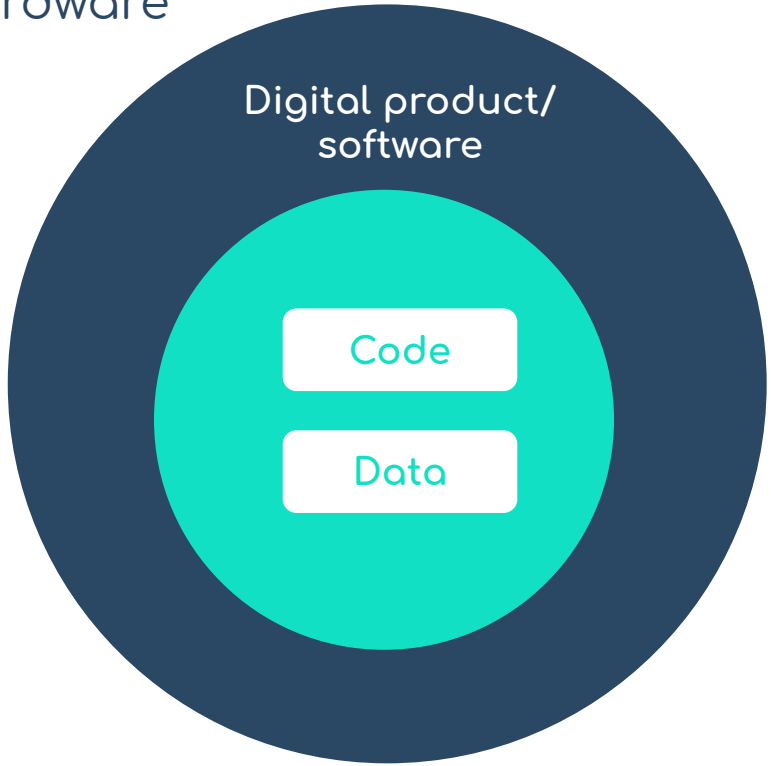
# HARDWARE

- Hardware = physical components of a computer
- The CPU (Central Processing Unit, prozessor) executes software
- The GPU (Graphical Processing Unit)
- Storage saves data
- Input: Mouse, keyboard, touchscreen ...
- Output: Monitor, sound, vibration ...
- Analogy: Hardware = skeleton



# SOFTWARE

- Software executes commands on hardware
- **Data** is stored information
- **Code** is text-based commands
- Analogy: Software = brain/nerves



# FUNCTIONS

- Reusable collection of commands

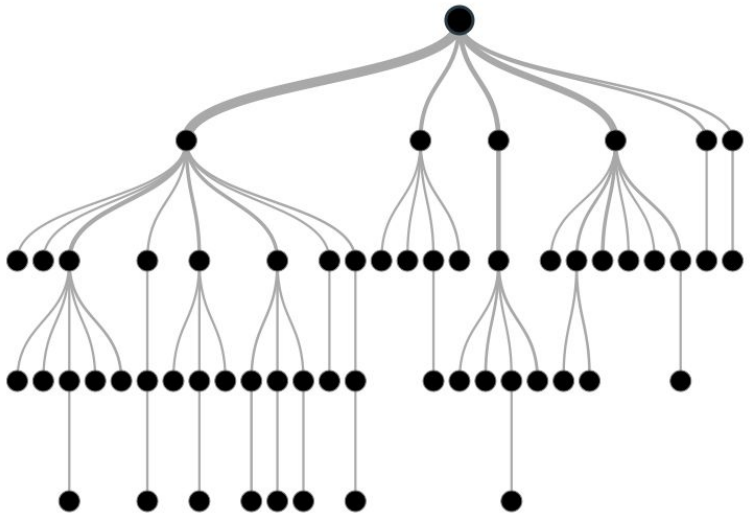
```
function roll (time, speed, dir) {
   setDirection(dir);
   setSpeed(speed);
}

function order(type, client) {
   var pizza = bake(type);
   deliver(pizza, client);
   …
}
```

# FLOW CONTROL

- Commands are executed when conditions are met.

If-else statement

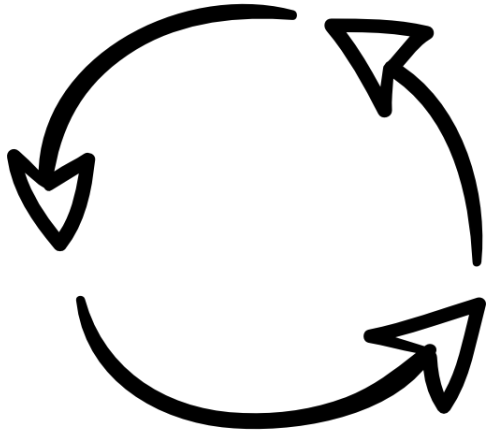
```
if (colour == green) {
    Sound(Boing);
}
if (Waiter.ready == Yes) {
    order();
}
```



# LOOPS

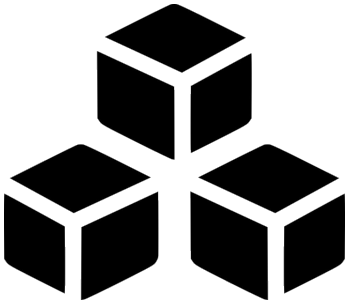
- Commands are repeated as long as condition is met

```
Var speed = 5;
while (speed <= 100) {
    speed = speed + 5;
    role (time, speed, dir);
}

while (orders < hunger) {
    order();
}
```



# LIBRARIES / MODULES

- Collection of functions and commands



# FRAMEWORKS

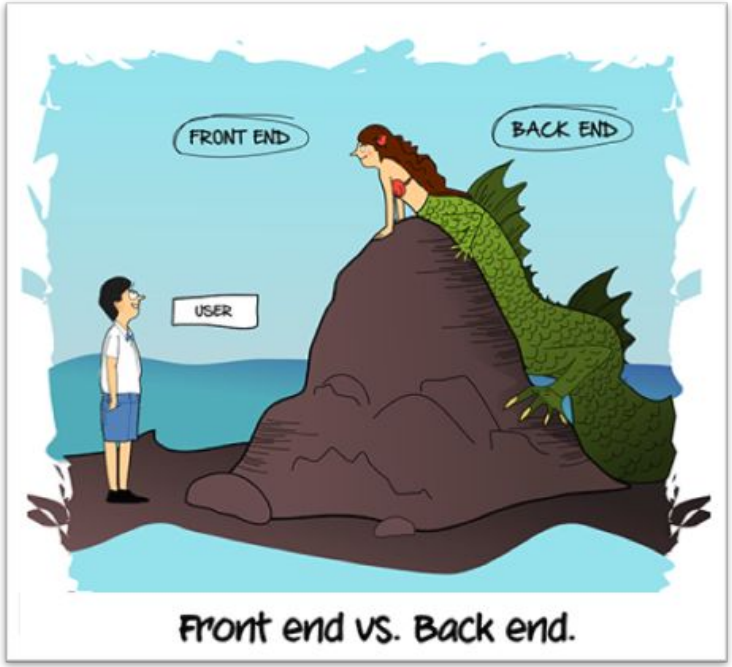
- Collection of functions, commands and rules
- Framework are dependent on language- and application ...

# SOFTWARE PHASES

- The software is developed/coded during the **development phase** (programmer's task).
- Once written, the software is executed at **runtime** (user task).
- Analogy: Car during manufacturing & during life cycle.

# FRONTEND & BACKEND

- Front-end and back-end describe layers of IT systems
- Front-end is closer to the user and to data inputs, can contain logic
- Back-end is closer to data processing
- Pair of terms is context-related
- Analogy: mimic as front-end, thoughts as back-end

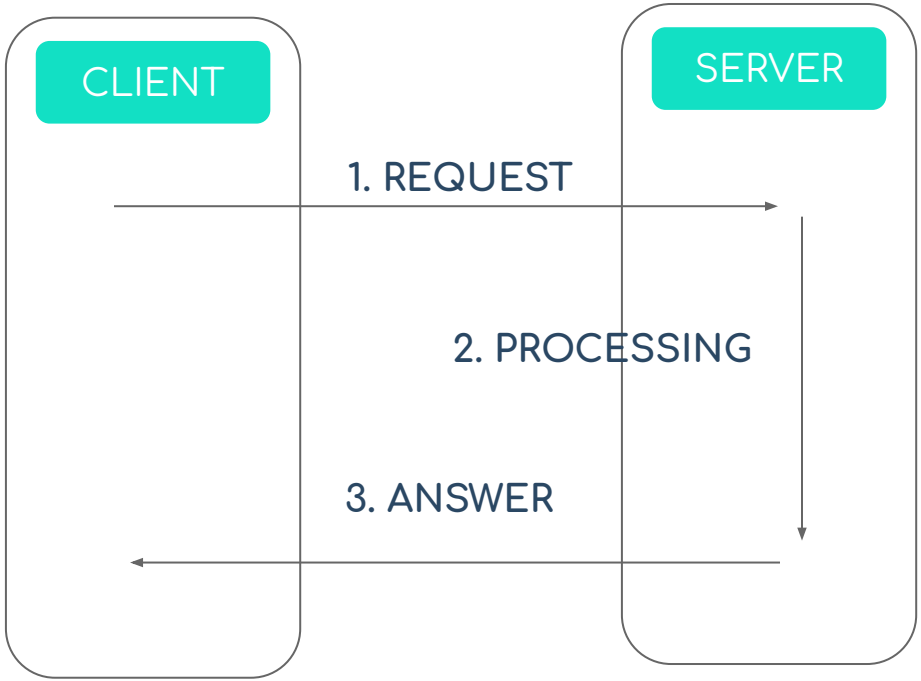


# INTERFACE

- Interfaces provide defined functions to for accessing the IT system from an environment.
- Graphical interfaces (also GUI, Graphic User Interface) are used by humans.
- API (Application Programming Interfaces) used by other programs
- Analogy: Power cable in socket, stove-plate and pot and lid

# CLIENT & SERVER

- Client-Server describes the distribution of tasks between two software systems.
- A physical device can be client and server at the same time.
- The term pair is context-related
- Analogy: Pizza customer is client, employee is server



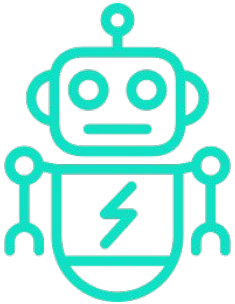
# LAYERS OF SOFTWARE DEVELOPMENT

- Software is built using programming languages.
- Modern software applications are built upon a stack of different tools and technologies.
- High-level programming languages, such as Python or JavaScript, offer layers of abstraction and provide meaningful concepts.
- Assembler code is very close to the hardware but is cumbersome to read and write.

# Agenda

1. <Hello World>
2. Computational Thinking
3. Theory & Terminology
4. Programming Robots
5. Coding & Languages
6. App Prototyping
7. You did it

# Roboter Coding



Sphero
Spark+

# Roboter Coding

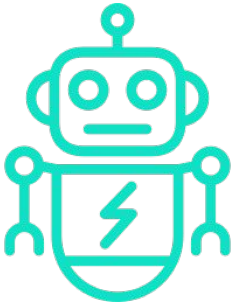
## WHY


In order to thrive in the 21st. Century promoting the right skills will play a major role in the success of individuals and corporates.

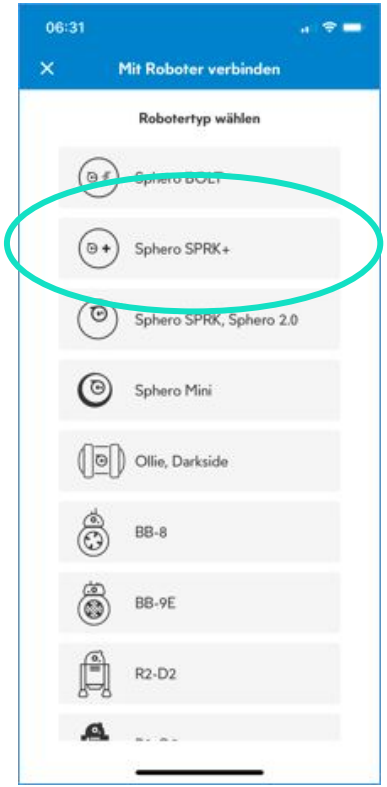
*"To be as productive as it could be, this new automation age will also require a range of human skills in the workplace, from technological expertise to essential social and emotional capabilities."*

McKinsey - Video:
The digital future of work:
What skills will be needed?

# Roboter Coding

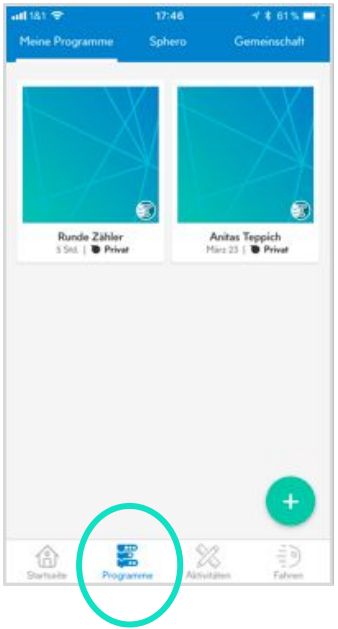


Sphero
Spark+



# Roboter Coding

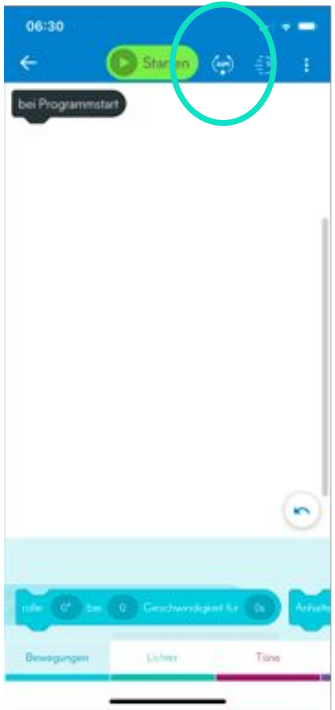
Click
Programmes



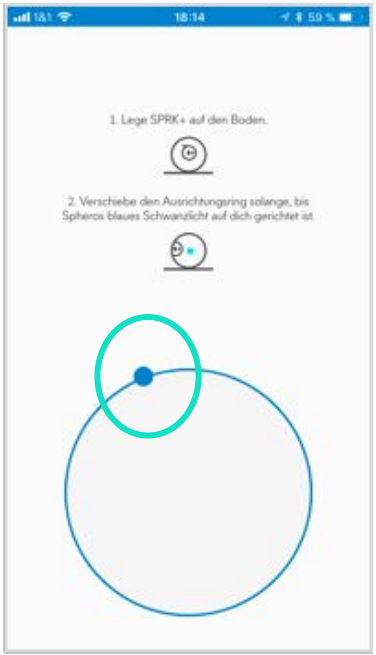

Click „+“ to create a new program and choose “Block“.

# Roboter Coding

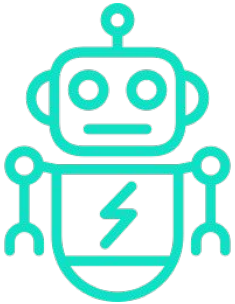
Click „Aim“ to callibrate robot.

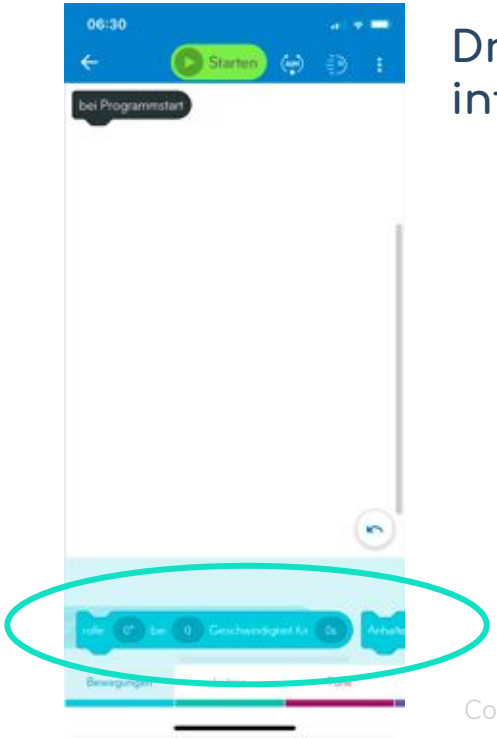



Rotate the blue dot on the smartphone so that your robots blue dot points to you.



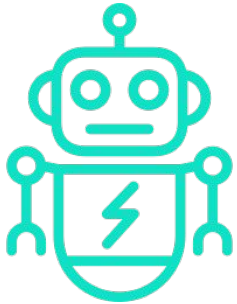
# Roboter Coding






Drag & Drop commands into the interface.

# The Parcours



- Turn the light to green.
- Drive along the course.
- After the first curve, change the color to red.
- Play a sound after impact.

# Agenda

1. <Hello World>
2. Computational Thinking
3. Theory & Terminology
4. Programming Robots
5. Coding & Languages
6. App Prototyping
7. You did it

# Coding Origins are Female

**Ada Lovelace**
**(1815-1852)**

Developed together with Babbage the preliminary stage to the first programming language.

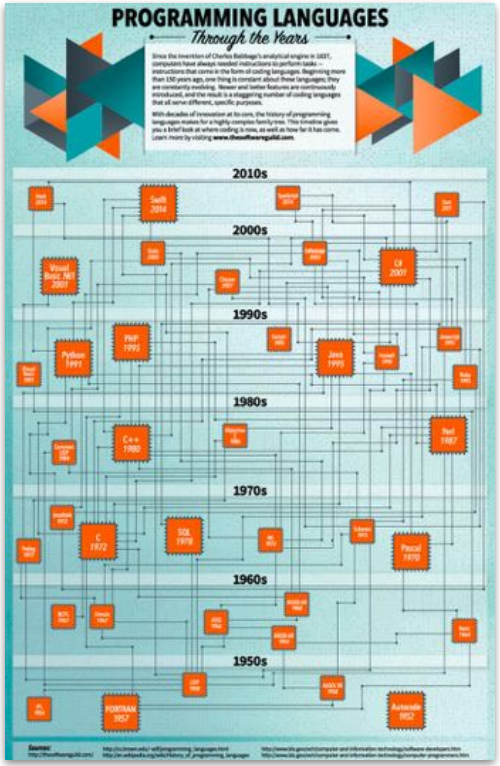
**Grace Hopper**
**(1906-1992)**

Involved in the development of the first compiler. This translated source code into a machine code understandable for the processor.

**Margret Hamilton**
**(1936-)**

Led the software team on the Apollo 11 mission. And she also established programming principles that are still valid today.

# Programming Languages




With decades of innovation at its core, the history of programming languages makes for a highly complex family tree.

This timeline gives you a brief look at where coding is now, as well as how far it has come.

# RUBY

```
MyLittleVar = "Hello World!"

5.times{puts MyLittleVar}
```

# PYTHON

```
MyLittleVar = "Hello World!"

for _ in range(5):

    print(MyLittleVar)
```

# JAVASCRIPT

```
var MyLittleVar = "Hello World!";

for (var x = 1; x <= 5; x++)

    {

    alert(MyLittleVar);

    };
```

# JAVA

```
class MyExample {
    public static void main(String[] args)
    {
        String MyLittleVar = "Hello World!";
        for (int x = 0; x < 5; x++)
        {

System.out.println(MyLittleVar);
        }
    }
}
```

# C/C++

```
#include<stdio.h>
int main() {
    char MyLittleVar[] = "Hello World\n";
    int x = 0;
    for (x = 0; x < 5; x++)
    {
        printf("%s", MyLittleVar);
    }

    return 0;
}
```

# ASSEMBLER

```
section .data
    MyLittleVar  db 'Hello World!', 10
    length equ $ - MyLittleVar;

section .data
 start:
    mov cx, 5 ; fill cx-register with 5
 loop:
    mov eax, 4 ; write(stdout, hello, length)
    mov ebx, 1
    mov ecx, MyLittleVar
    mov edx, length
    int 80h

    loop schleife ; jump to 'loop' as long as cx > 0
and decrease cx by 1
    mov ebx, 0  ; Call: exit
    mov eax, 1
    int 80h
```

# MACHINE CODE

```
01001000 01100101 01101100 01101100 01101111 00100000
01010111 01101111 01110010 01101100 01100100 0001010
01001000 01100101 01101100 01101100 01101111 00100000
01010111 01101111 01110010 01101100 01100100 0001010
01001000 01100101 01101100 01101100 01101111 00100000
01010111 01101111 01110010 01101100 01100100 0001010
01001000 01100101 01101100 01101100 01101111 00100000
01010111 01101111 01110010 01101100 01100100 0001010
01001000 01100101 01101100 01101100 01101111 00100000
01010111 01101111 01110010 01101100 01100100
```

=

Hello World Hello World Hello World Hello World Hello World

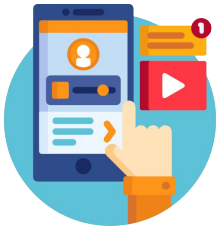
# Let's Play

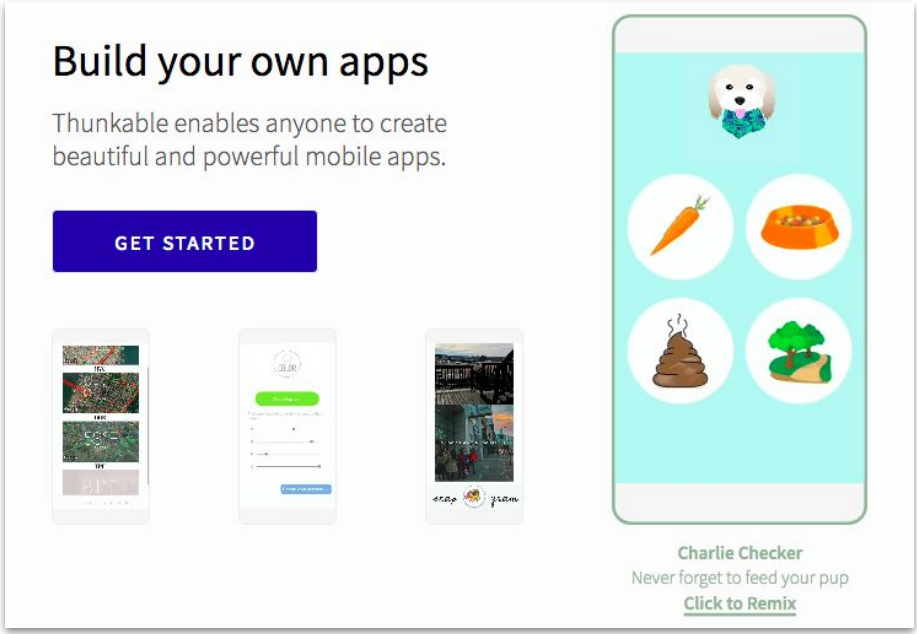
# Agenda

1. <Hello World>
2. Computational Thinking
3. Theory & Terminology
4. Programming Robots
5. Coding & Languages
6. App Prototyping
7. You did it

# App Prototyping

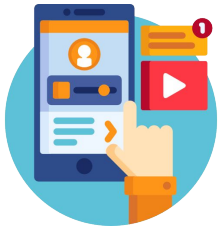
## thunkable





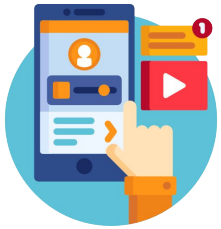

- Block-based visual programming language.
- Lets users create programs by manipulating program elements graphically rather than by specifying them textually.

# App Prototyping



- Go to www.thunkable.com and sign up
- Download Thunkable App
- Open the Thunkable Live app and log in
- On your computer, click the "Live Test" button
- When you make changes to your app on the computer, they will update on your mobile device.

# App Prototyping



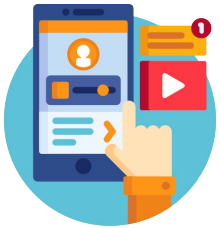
DESIGN

Here we will insert the components.

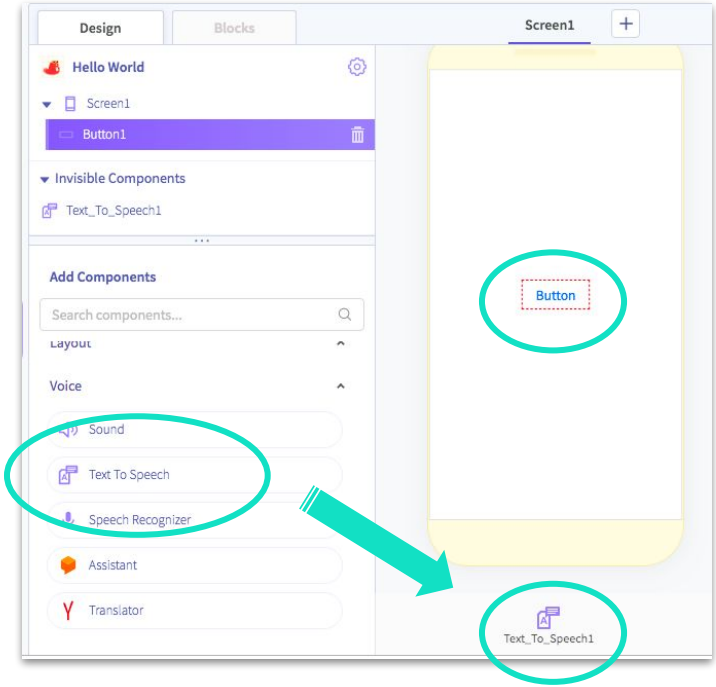
BLOCKS

Here we will code using block-coding.

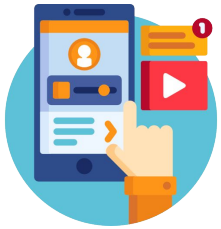
# App Prototyping



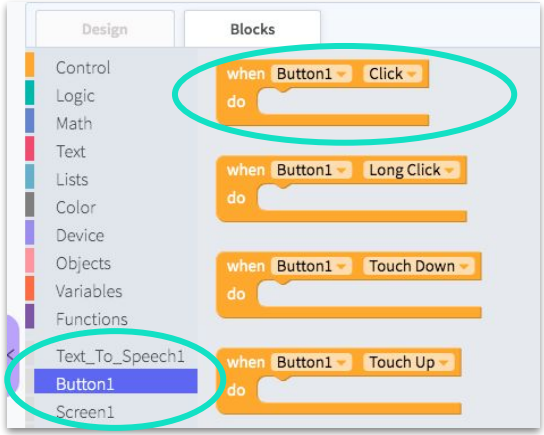
- In the Design-Area add the following two components: Buttons & Text to Speech



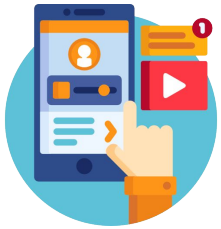
# App Prototyping



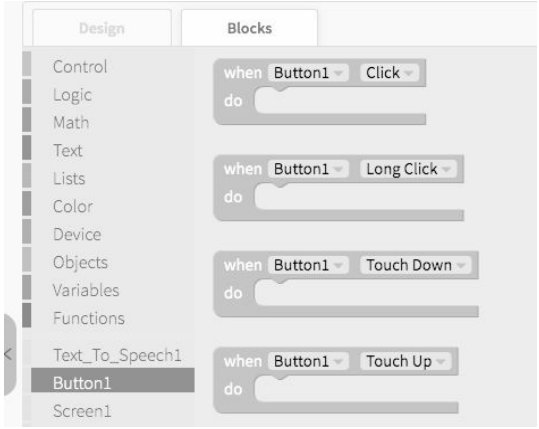
- In Block-Area on the left side click on Button1.
- Drag & drop the when Button1 Click function onto screen.

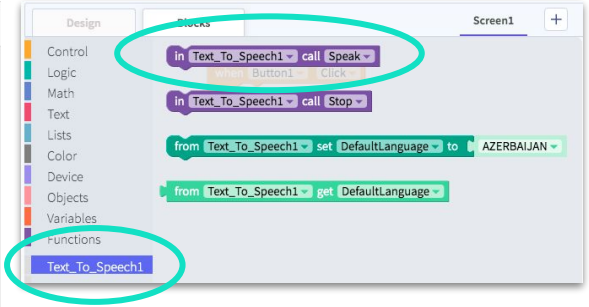


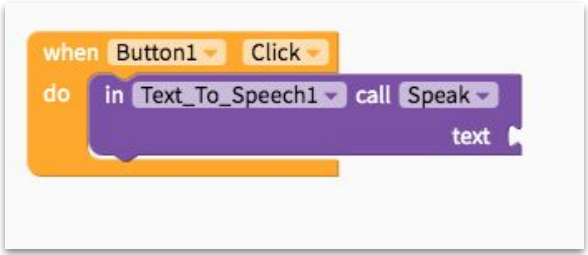
# App Prototyping



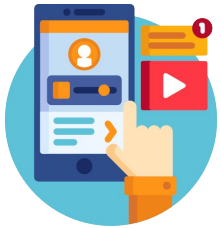
- In Block-Area on the left side click on Text_To_Speech1.
- Drag & drop the in_Text_To_Speech1 function into yellow brackets.



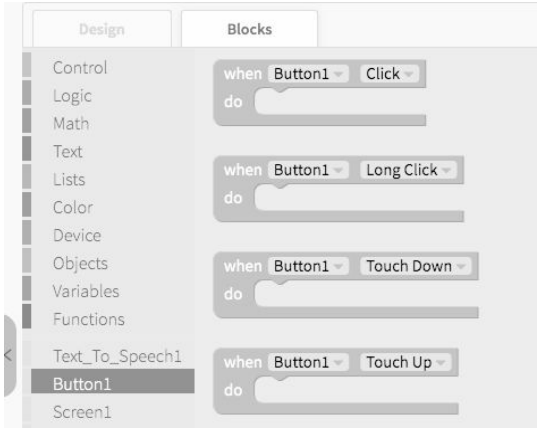



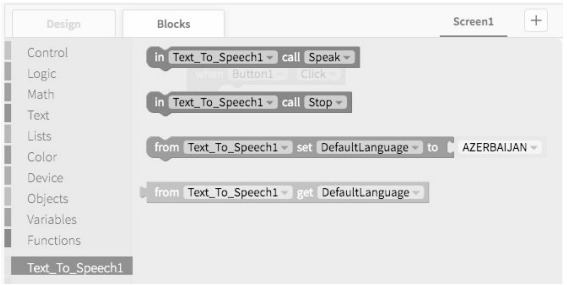



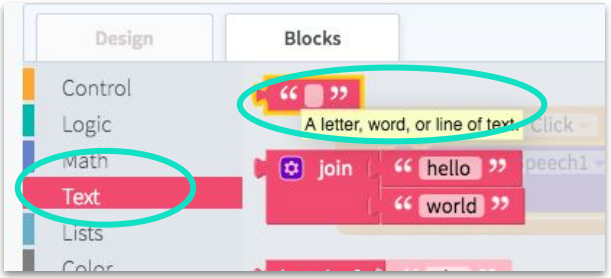
# App Prototyping



- Im Block-Modus click on Text and choose field with “ ”
- Add next to purple text funtion



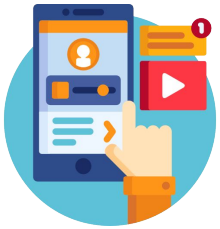



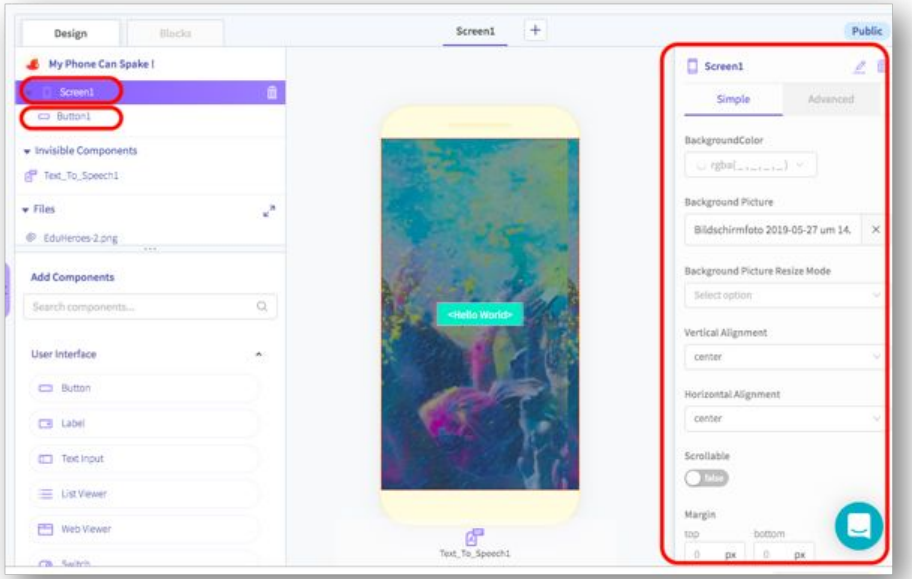





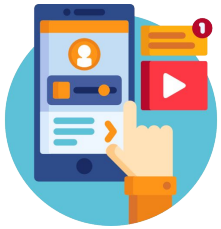
# App Prototyping

- And now add some color.





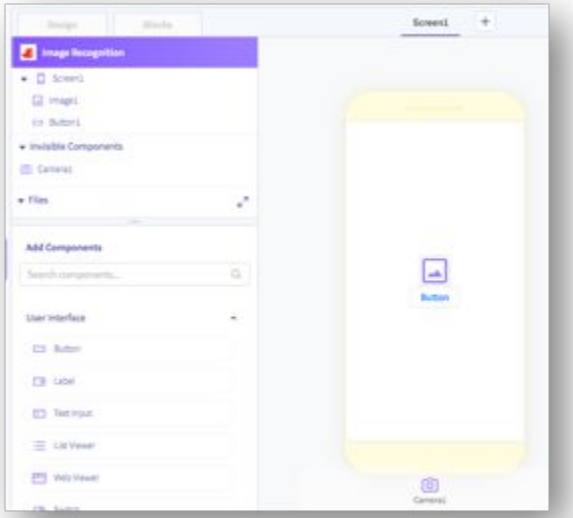
# APP PROTOTYPING



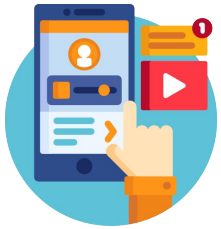
- Let's build an AI-based app

In the Design area add the following components:

- Button
- Image
- Camera



# APP PROTOTYPING



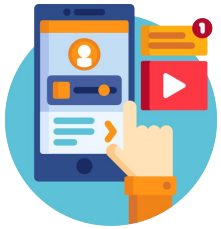
In the Design area add the following components:

- Button
- Image
- Camera

In the Block area:

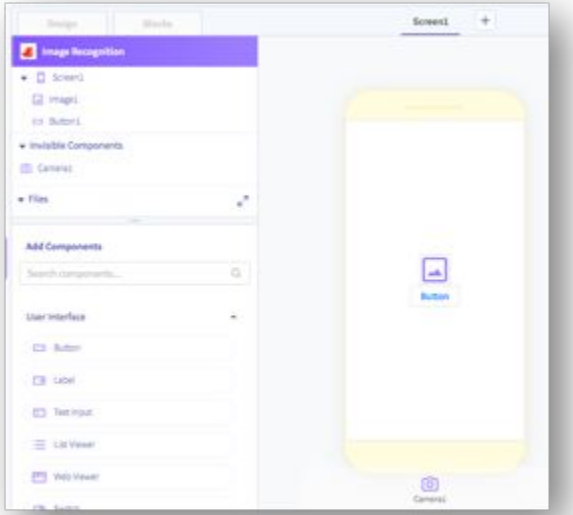
1. Button Function:
   When Button1 Click
2. Camera Function:
   In Camera1 Call Take Photo
3. Image Function:
   Image1 set Picture to
4. Move Photo to “ ! “

# App Prototyping



In the Design area add the following components:

- Button
- Image
- Camera



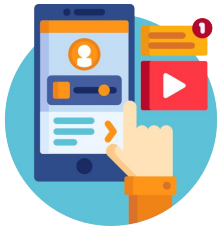

In the Block area:

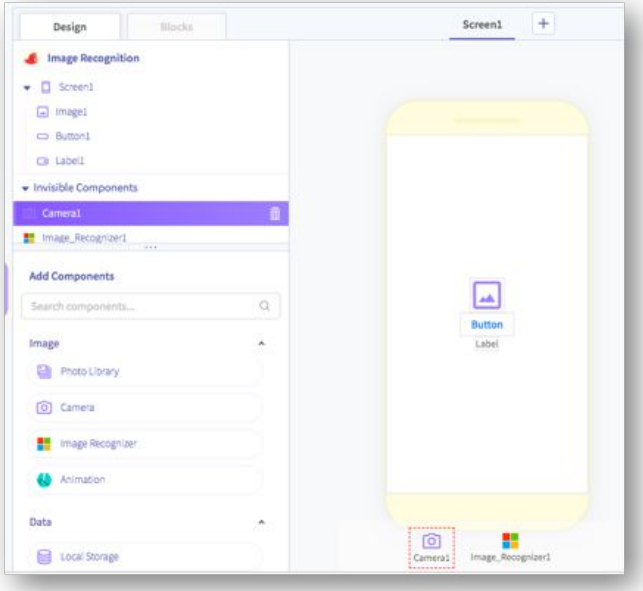
1. Button Function:
   **When Button1 Click**
2. Camera Function:
   **In Camera1 Call Take Photo**
3. Image Function:
   **Image1 set Picture to**
4. Move Photo to “ ! “


Open Thunkable App and test.

Test to see if the image on the screen is set to the picture that you took with the camera.

# App Prototyping

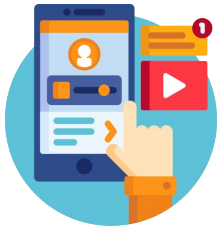





Im Design area:

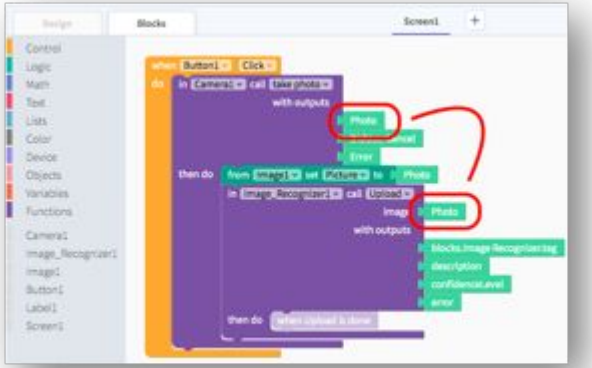
- Add component Label
- Move Label under Button
- Add component Image Recognizer

# App Prototyping

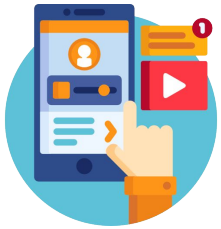


In Block area:

- Click Image_Recognizer1
- Drag & drop Image_Recognizer1 call Upload into the in Camera1 call TakePhoto bracket
- Drag and drop the "photo" block from the Camera1 block into the "image" socket on the Image_Recognizer1 block.

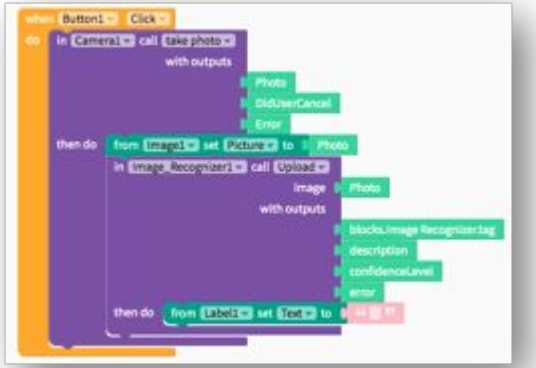


# App Prototyping

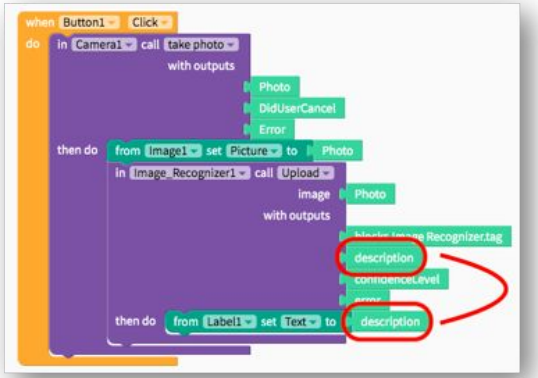


In Block area:

- Open the drawer for Label1.
- Drag and drop from Label1 set Text to block inside the in Image_Recognizer1 call Upload block.



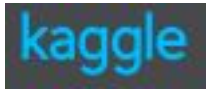

- Drag and drop the "description" block from the Image_Recognizer1 block into the opening of the "from Label1 set Text to" block.



# Software check (optional)

# Coding Resources

## Online






# Coding Resources

## Bootcamps




## Communities



## Fun


# Agenda

1. <Hello World>
2. Computational Thinking
3. Theory & Terminology
4. Programming Robots
5. Coding & Languages
6. App Prototyping
7. You did it