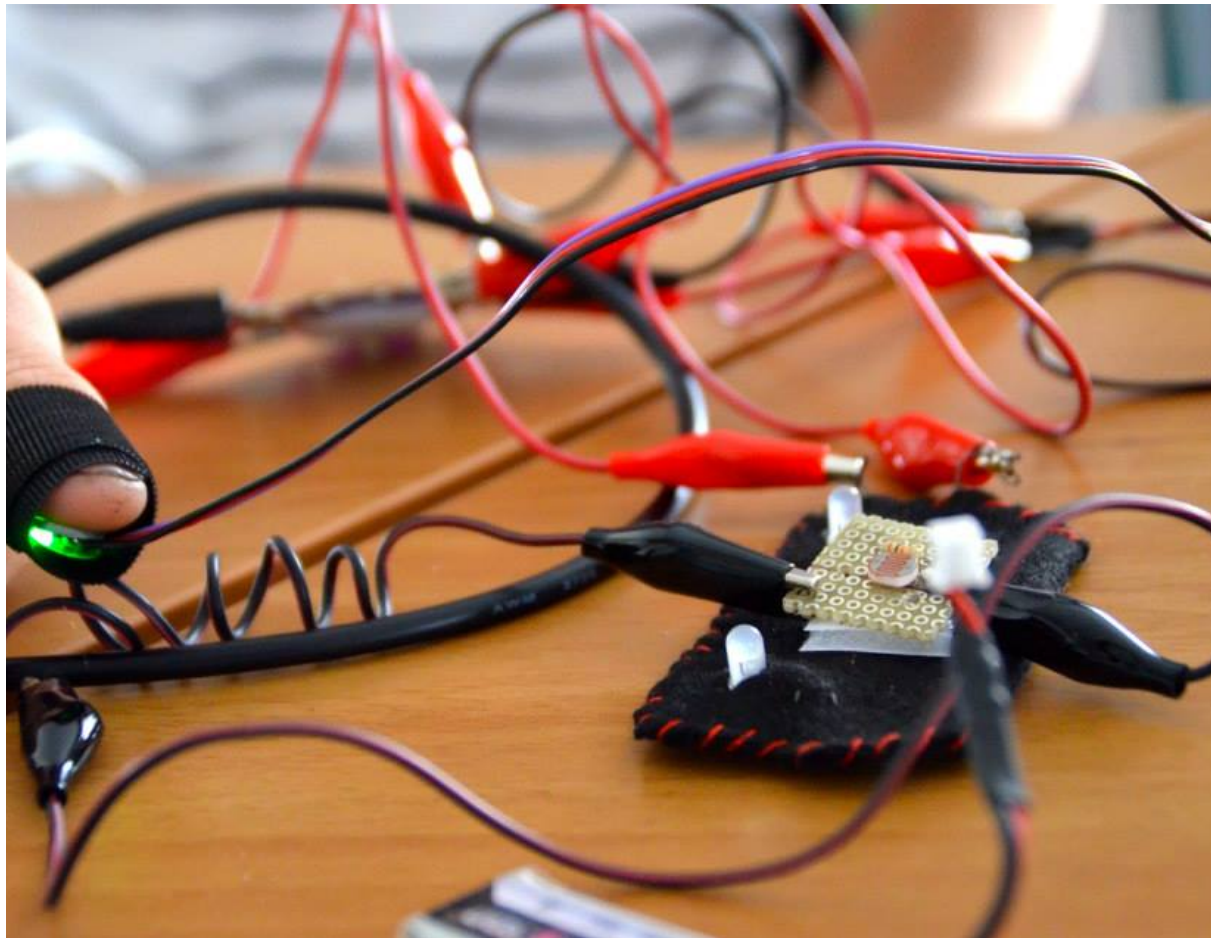


IOT's for educational and school aims

The task of this manual is to introduce different methodologies, Learning Scenarios and educational activities on coding and IoT in education.





To thrive, they must learn to design innovative solutions to the unexpected problems that will undoubtedly arise in their lives. Their success and satisfaction will be based on their ability to think and act creatively. Knowledge alone is not enough: they must learn how to use their knowledge creatively

- Mitchel Resnick, MIT Media Lab

An introduction to Internet of things and wearable device in Education

In this manual we can learn how to use some of simple online editor to program and interact with simple sensors and outputs. Internet of things is an increasing field of the market, from thermostats to smart watch.

In the field of education it will be important to introduce all this technology because for student it is an engaging activities and to apply the theory of 4P.

Project, Peer, Play and passion an methodology that is introduced by Mitchel Resnick from MIT Lifelong Learning Lab (MediaLab) and it suit very well for IoT educational activities. Moreover in this manual we present the possibility to create some program with some sensors present in different commercial platform all link to the online editor scratch3. In Learning scenarios we will use also different program like Snap for Arduino or makecode. All used software are free and compatible with the most important robotics platform like Lego, Microbit, Arduino and Raspberry PI.

Introduction to Arduino and similar shield

We choose to present only the Arduino shield and compatible platform because that are open source and in EU project we consider ethical the use of cheap and open source platforms like Arduino and Elegoo.

These shield are totally compatible with a plethora of sensor and actuators.

Arduino is the most popular open source electronics platform that change the world of education, thanks to the low price and the easy use of prototyping and programming.

A typical shield is composed by a 8 bit's micro controller with different chips from Mega AVR family. For each shield there Digital Input, Digital Output and analogic input and output.

Free online software

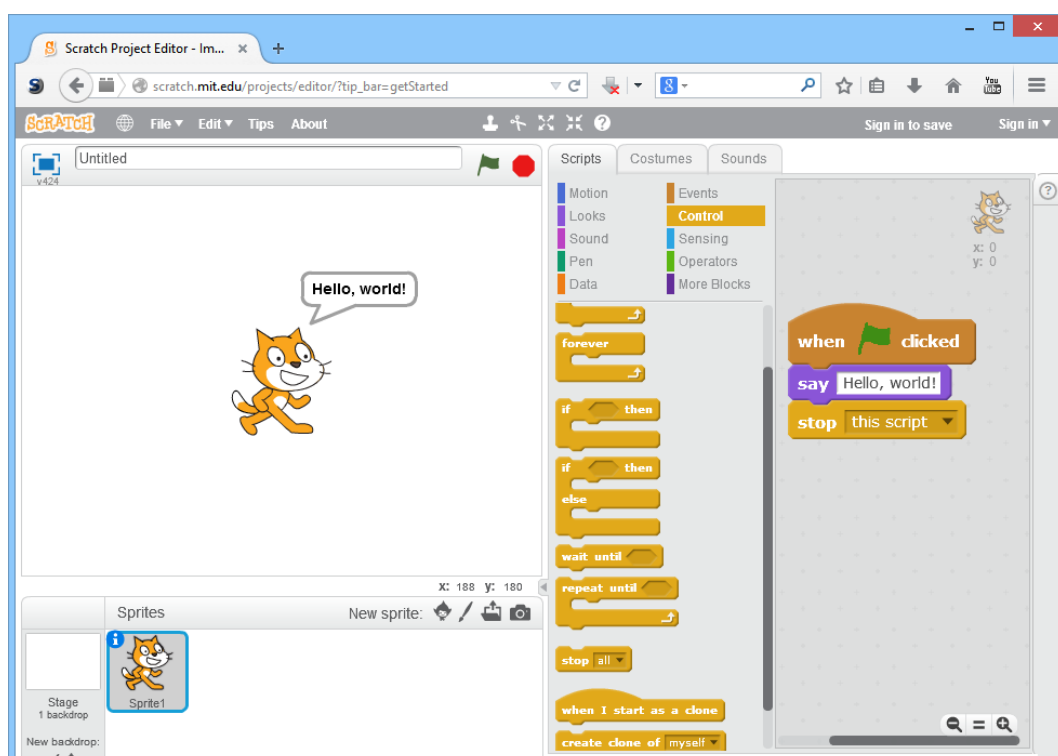
In this project we choose to use only free software to program our objects. The most popular program is Scratch, now in his version Scratch3.

Scratch

Scratch is a visual programming editor developed by MIT Media Lab. Scratch is born in the 2006 and now is used in most schools in the world. It is available in more than 70 languages.

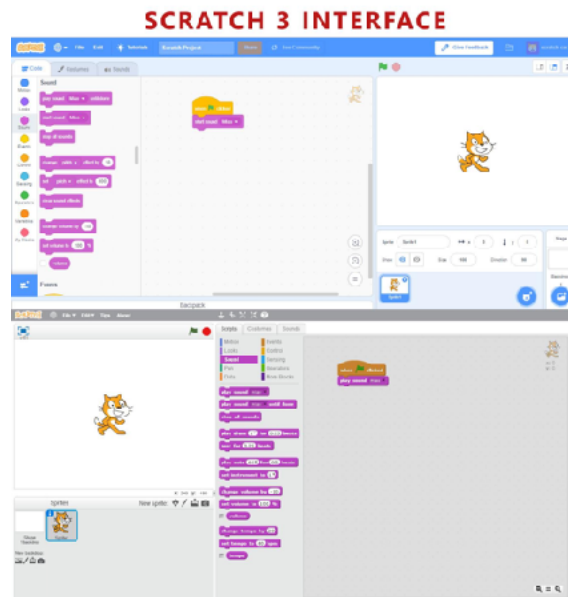
Scratch 2 is available also in offline mode. You can download it here <https://scratch.mit.edu/download>. Scratch 2 is not available on tablets.

Those are the main differences between Scratch 2 and 3. Both versions of scratch can work online through internet browsers on pc. Scratch 3 is builded in HTML5 so it can works also on tablets Android or ipads, but it hasn't an offline version as opposed to Scratch 2 that is bulded in Flash. Both programs can interact with external devices but only on computers because it need to install a little linker program. Scratch 2 can natively control Lego WeDo 1 and 2 and Picoboard. Scratch 3 can natively control Lego WeDo 2 and Lego Mindstorm EV3, Microbit, it has also extra functionality like text to speech in different language and translator. It is possible to build new extensions so in the future will be more extra functionality. There are some others extra functionality common to the two platforms like drawing pen, musical instruments and video sensing.



Main differences between Scratch 2 and Scratch 3		
	Scratch 2	Scratch 3
Offline version (Windows, OSX)		
Online version	(only on computer)	(on computers and tablets)
External devices		
Lego WeDo 1		
Lego WeDo 2		
Picoboard		
Lego Mindstorm EV3		
Microbit		
Extra Functionality		
Text-to-speech in different languages		
Drawing Pen		
Musical Instruments		
Video sensing		
Translate text		

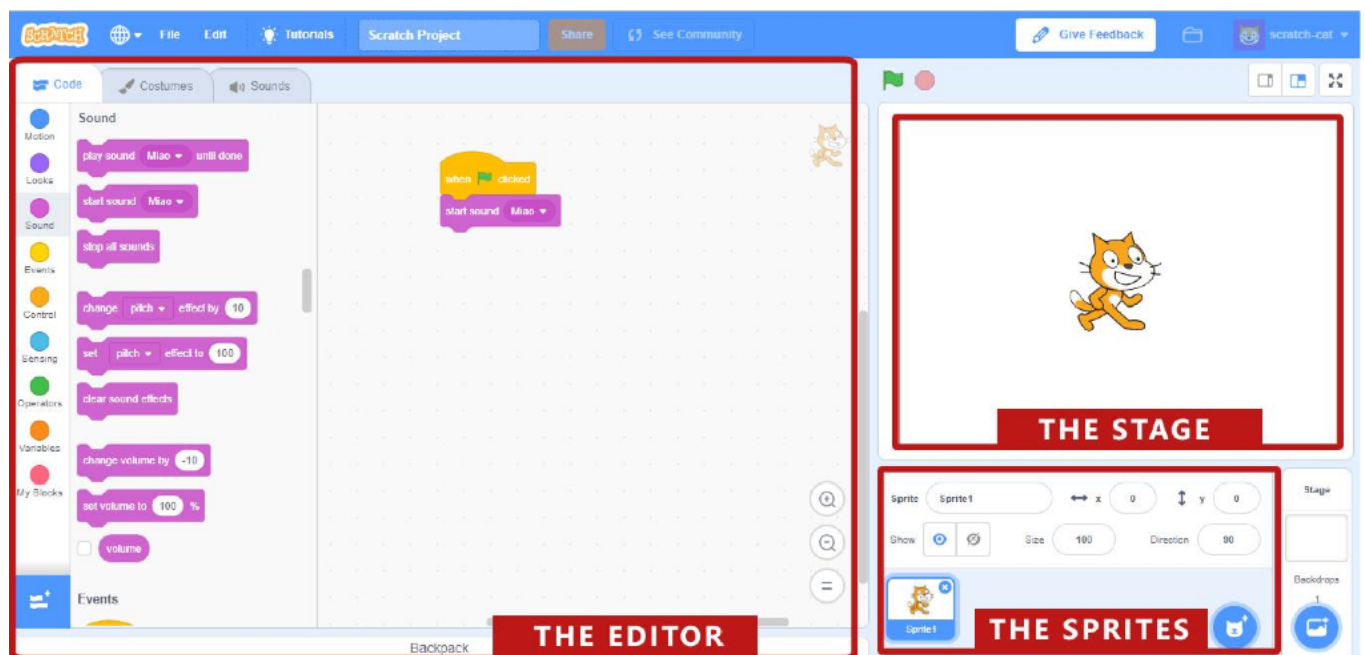
Both versions allow to save and share Scratch projects on the Scratch Community. The files are compatible with the both versions. Since January 2019 the only online version will be Scratch 3 and the only offline version will be Scratch 2.



SCRATCH 2 INTERFACE

The interface

The new interface of Scratch 3 is showed in the following picture.



It is divided in three main areas:

The stage is the most important area where the coding program will 'come to life'

The sprite's area, in this areas there are all the components (sprite) who are part of the stage

The editor area, it comprises three different types of editors.

The code editor: contains the list of available blocks and all the blocks used to describe the respective sprite's behavior. It is important to understand that every sprite has its own blocks, so when we will select different sprites in the sprite's area the blocks in the script area will change.

The costumes editor: permits to draw and modify the aspect of the sprites. Every sprite has its own costumes.

The sound editor: allows to record and edit sounds to use in the program.

Snap for Arduino

Snap for Arduino is a modification of the blocks program Snap!, created by the University of California at Berkeley. Thanks to Snap (that is continuously developed) we can program easily all Arduino boards.



From the official website, the features of Snap4Arduino are:

- Blocks-based, dynamic, live, concurrent, parallel programming
- Almost all Arduino boards supported
- Uses standard Firmata firmware
- Auto-configurable pinouts and high level hardware abstractions
- You can interact with multiple boards at the same time
- Desktop-based versions for the three major OSes
- Online version that can connect to Arduino boards via a Chrome/Chromium plugin
- Free software licensed under the Affero GPLv3
- Transpilation of simple scripts into Arduino sketches
- HTTP protocol for remote control and live-streaming of the Snap! stage

- Command line version for embedded GNU/Linuxes

How to install each software

To install Scratch 3: <https://scratch.mit.edu>

To install Snap4Arduino: <http://snap4arduino.rocks>

Focus on Scratch 3

Scratch is designed especially for ages 8 to 16, but it is a very useful tool for any programming beginner.

Scratch is used in more than 150 different countries and available in more than 40 languages.

Scratch is used as the introductory language because the creation of interesting programs is relatively easy, and skills learned can be applied to other programming languages such as Python and Java.

Category	Notes	Category	Notes
Motion	Moves sprites, changes angles and changes X and Y values.	Sensing	Sprites can interact with the surroundings the user has created
Looks	Controls the visuals of the sprite; attach speech or thought bubble, change of background, enlarge or shrink, transparency, shade	Operators	Mathematical operators, random number generator, and-or statement that compares sprite positions
Sound	Plays audio files and effects. Programmable sequences are now available as an extension category named "Music".	Variables	Variable and List usage and assignment
Events	Contains event handlers placed on the top of each group of blocks	My Blocks	Custom procedures (blocks).
Control	Conditional if-else statement, "forever", "repeat", and "stop", etc.		

In addition, Scratch includes the following extensions:

- Music
- Pen

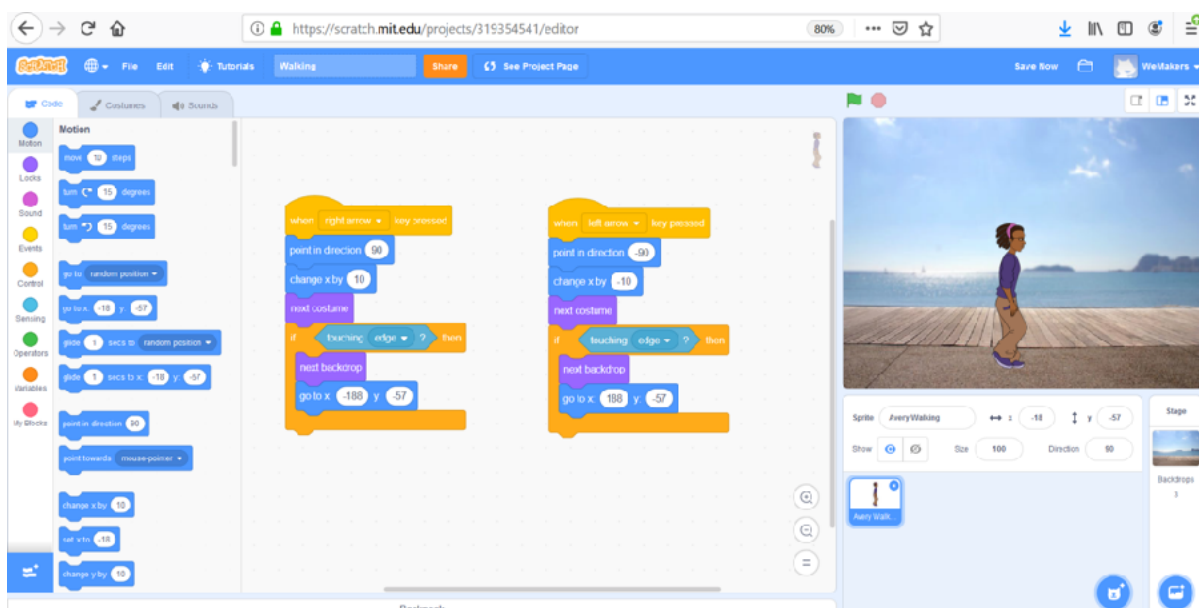
- Video Sensing
- Text To Speech
- Translate
- Makey Makey
- micro:bit
- LEGO MIDSTORMS EV3
- LEGO BOOST
- LEGO Education WeDo 2.0
- Go Direct Force & Acceleration

Application 1: a character who walks (left/right arrows)

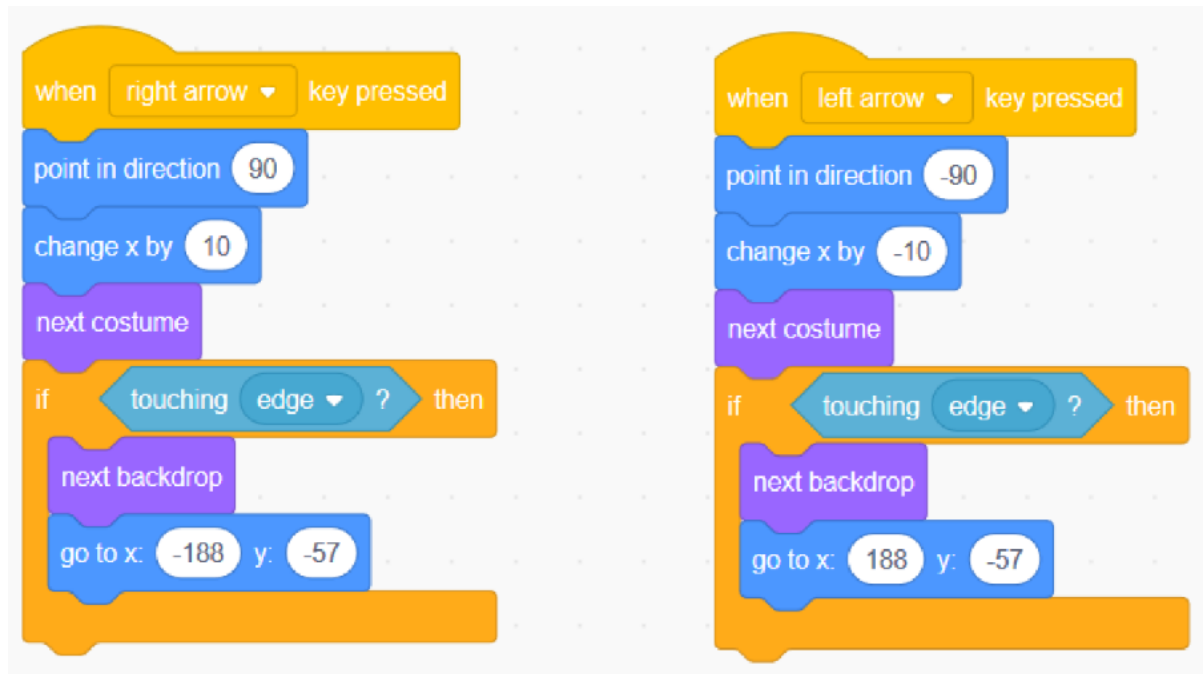
The following application has a character (sprite) controlled by left and right arrows, who walks in three or more backdrops (backgrounds).

How to program step by step:

1. Choose a sprite (preferably one with walking/flying/swimming costumes)
2. Choose three or more backdrops



3. Write the scripts presented in the next image
4. Use the right and left arrow to test the result



Challenges Proposal

Challenge 1: Make the character jump and advance when up arrow pressed.

Challenge 2: Create a game:

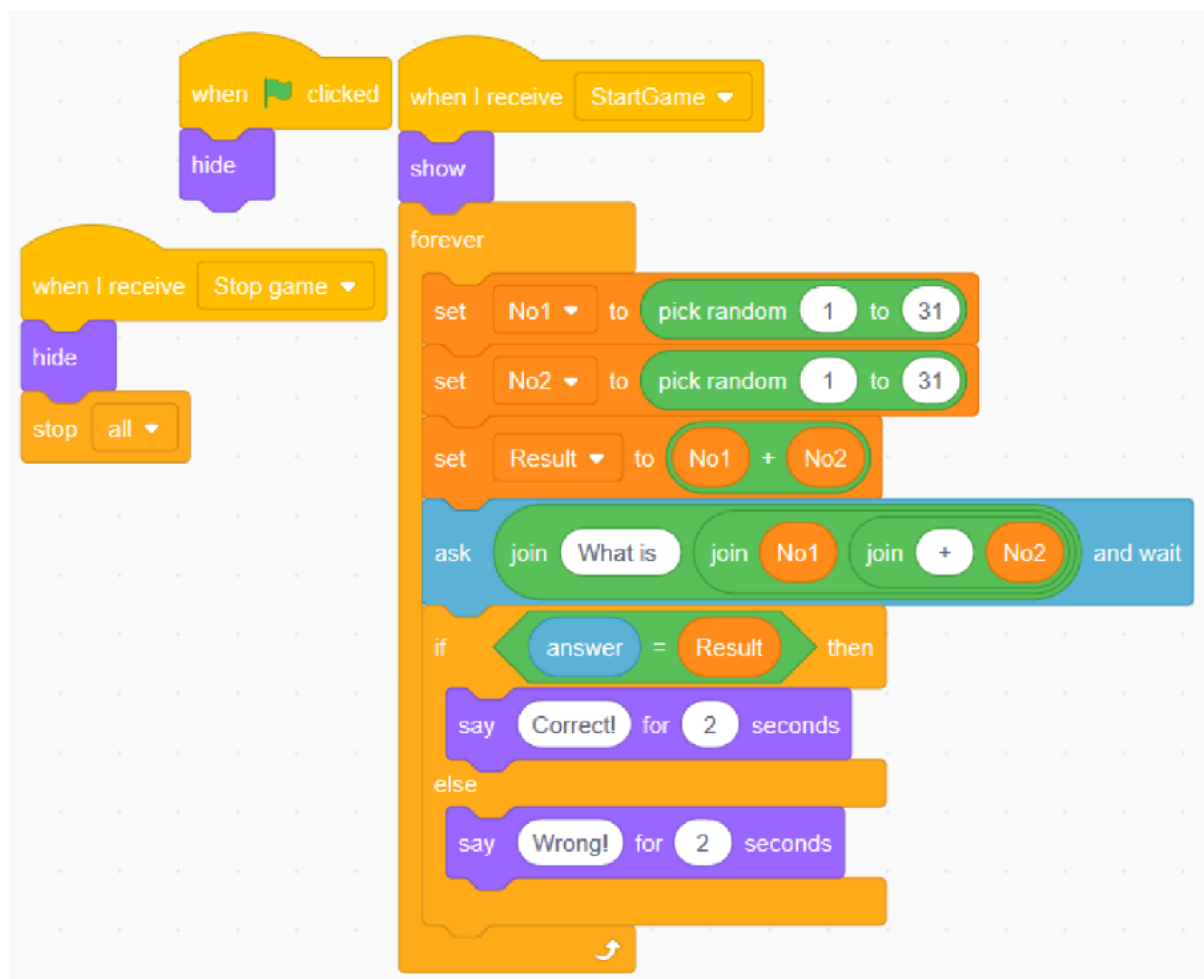
1. 1 controllable character and 1 frog who advance randomly (only x value change)
2. The character must not touch the frog (jump over the frog)
3. It has 3 lives
4. If it touches the frog it loses a life
5. Game ends when no life left

You may add the appropriate code to make the character change to a frog or other animal for 2 seconds when touches the frog.

Challenge 3: The game generates random additions and display *Correct* or *Wrong* to each answer. It has three backdrops (2 buttons and a character who put de question and give the answer).

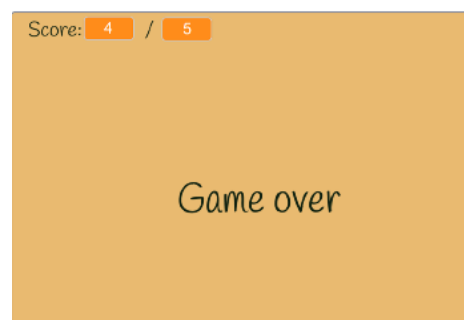


The scripts for each sprite and for the stage are:





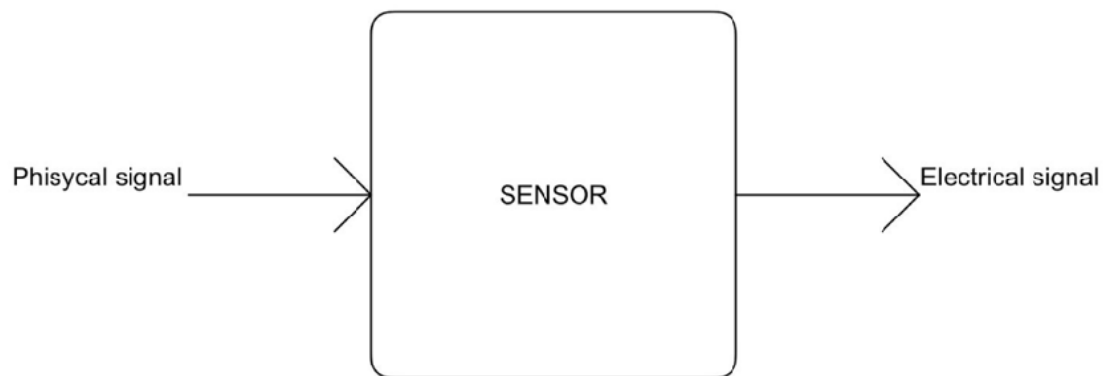
Challenge 4: Add two variables to count the total number of questions and the total number of correct answers. Show the two values at the end.



Challenge 5: Create your own quiz game with 2-3 questions.

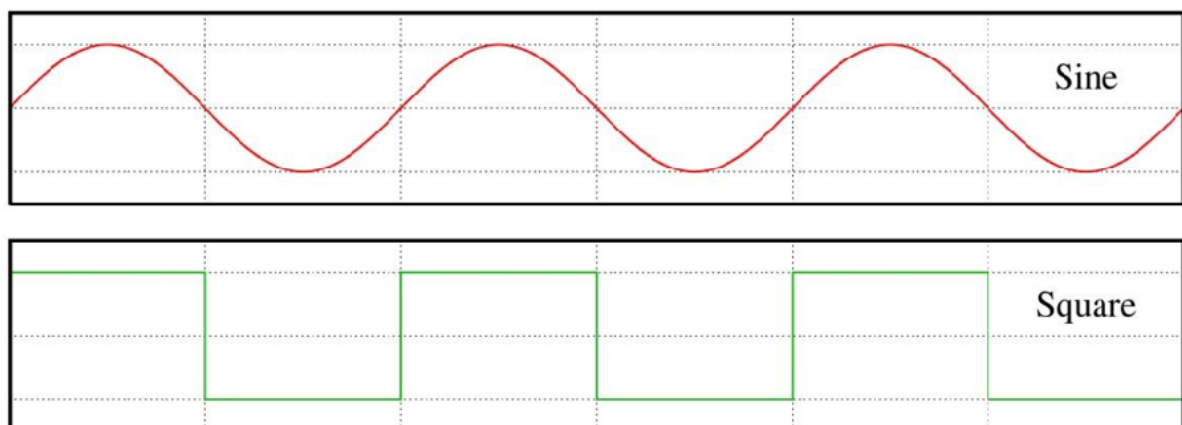
Sensors: a short overview

A sensor is a device that is used in electronics to detect changes of a physical parameter in the environment and send this information, codified into an electrical signal, to other devices to be manipulated and analyzed.



The electrical signal could be analogical or digital. Analogical means that the signal varies with continuity, getting all values between a minimum and a maximum. Digital means that the signal could assume only a limited number of values, typically two: low and high level.

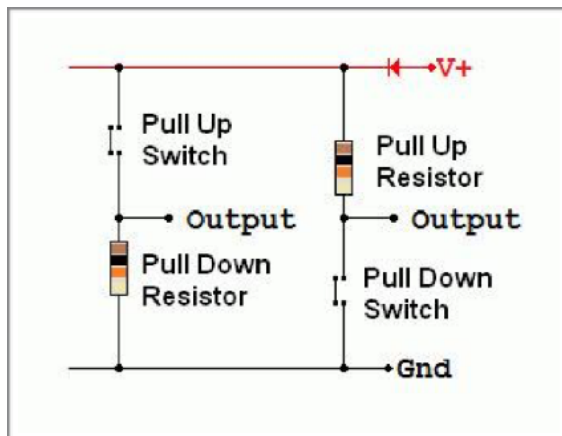
You can see in the below figure two examples of signal: a sine wave and a square wave. The first one assumes all values from the lower to the upper bound, so it is an analog signal. The second one assumes only two values, stepping from a low state to a high one, and vice-versa, so it is a digital signal.



An example of analog signal is the temperature: it could assume every value from the absolute zero to infinity. The typical example of digital signal is the state of a button: it could be only pressed or depressed. The typical analog sensor is the variable resistor (for example the potentiometer to turn up and down the volume of our stereo amplifiers), and the digital one is the button, that we will see in the next section.

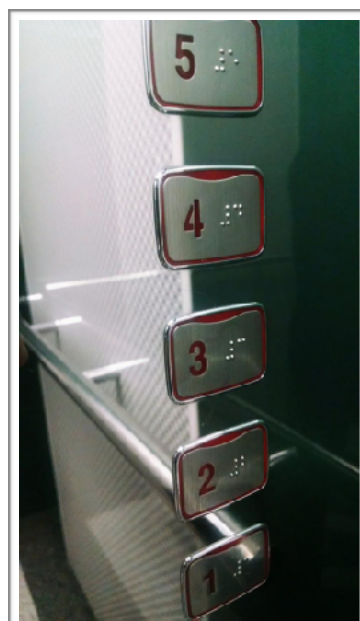
Touch sensor

The simplest sensor that we can construct is the touch sensor. Does exist different types of touch sensor, but the simplest we can use is the button and the Interruptor. A button is only an electromechanical device which is able to close or open an



electrical circuit. When the circuit is closed, the electrical current could flow, otherwise when it is opened the current could not flow. Putting a button in an proper circuit with a resistor we obtain a circuit that could give only two voltage values, low or high, corresponding responding to the state of the button. You can see the below schematic for connections. You can put the resistor and the switch in two ways. The first one, with the “pull down resistor” offer

a low value on the output when the switch is open and an high one when it is close. The “pull up resistor” version offer the opposite behaviour. For example we use buttons and interrupters in our home to turn on and off the lights, or in the elevator to select the right floor.

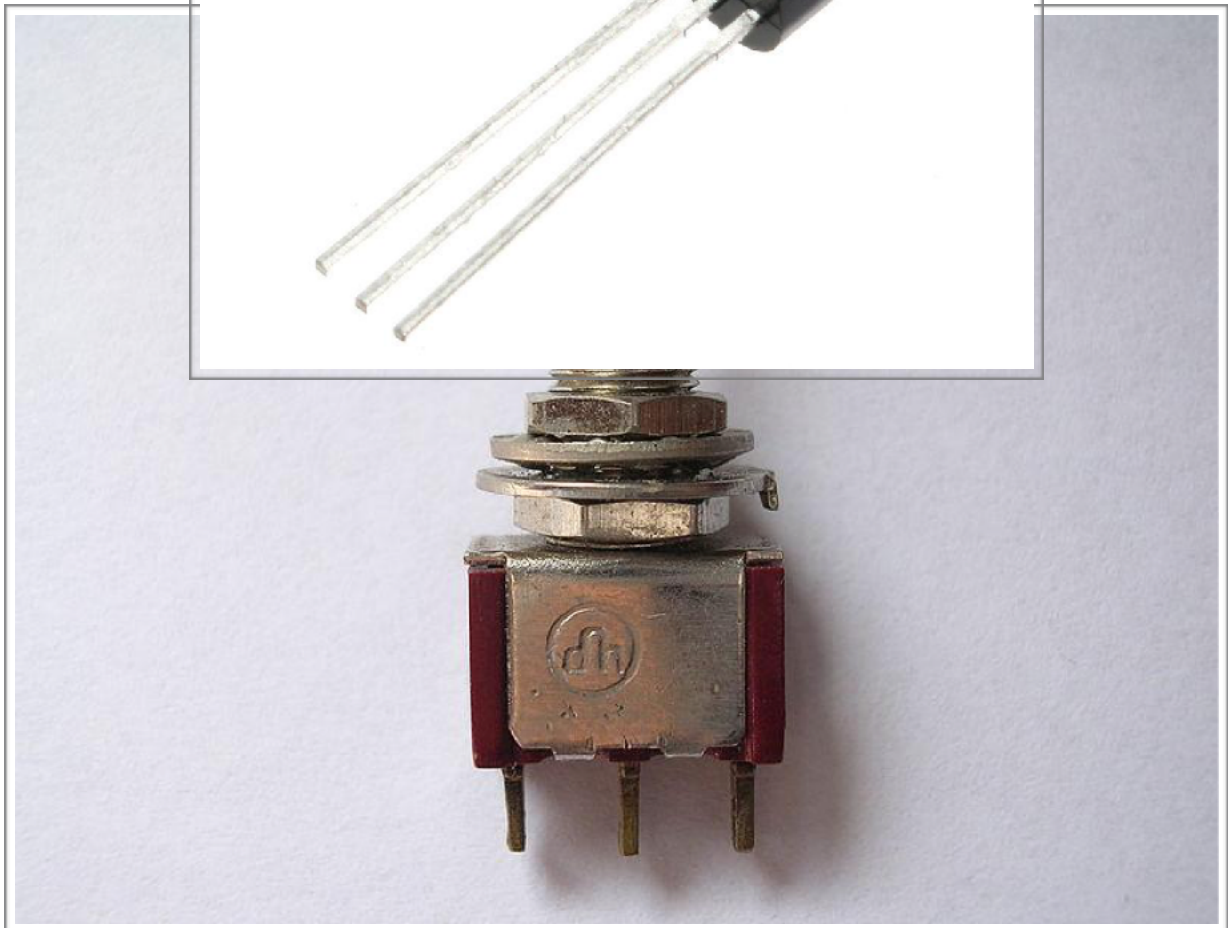
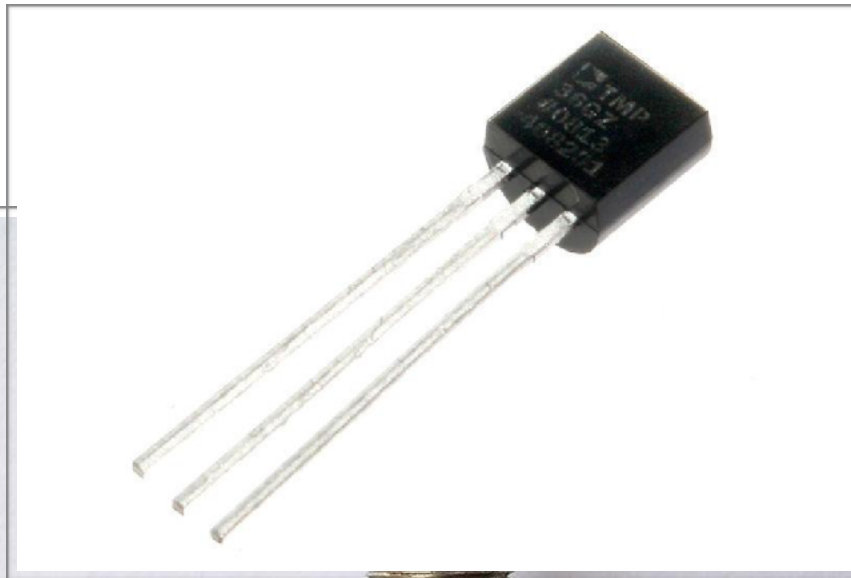


At this link you can find more information about button and an open source programmable board: <https://www.arduino.cc/en/Tutorial/Button>

Does exists other touch sensors, and the most popular are the capacitive sensors. They're technology is used in our touch screen devices. They measures the variation of the capacity of a capacitor due to human presence. At the end, we have a device

offer a
output,
button.

which
two level
like the

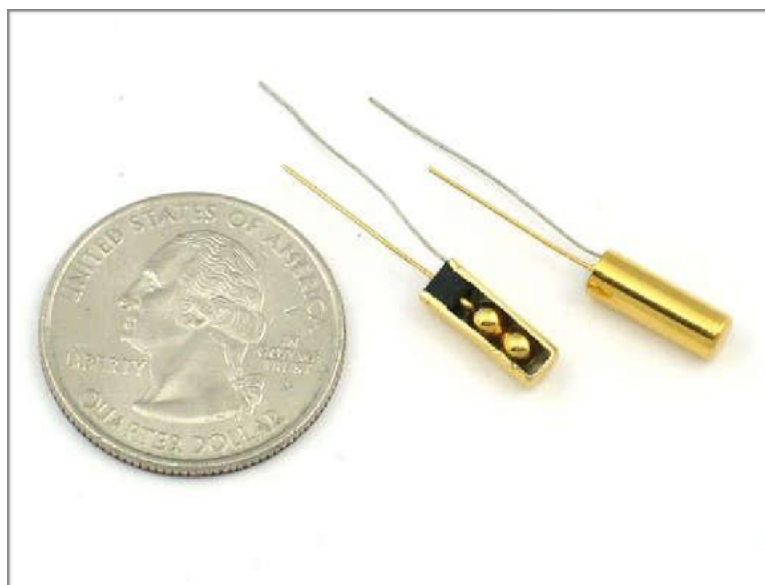


Tilt sensor

The tilt sensor is used to detect if an object is tilted in one, or more, direction. The easiest tilt sensor is the mercury switch: it is composed by a vacuum bulb containing two contacts and a little amount of mercury, free to move. When the bulb is tilted the mercury moves and when it reach the contacts it closes the circuit. The mercury is often replaced with a metallic ball. Using more contacts it is possible to create multi-axis tilt sensors.

A tilt sensor is used, for example, in some light emitting yo-yo which flashes when playing or in simple pitch-roll sensor in some vehicles.

At this link you can find an example with a single axis tilt sensor and a programmable board: <https://learn.adafruit.com/tilt-sensor/using-a-tilt-sensor>

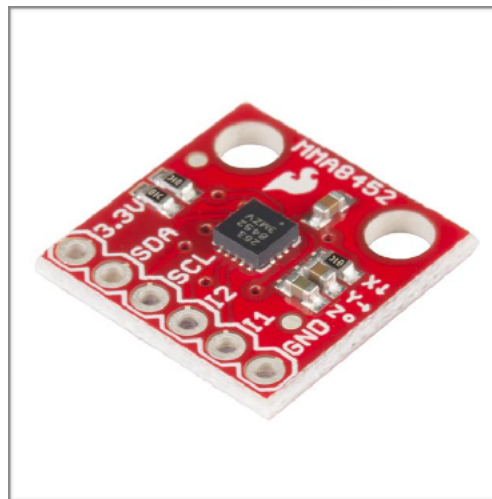


Accelerometer

The accelerometer is a device used to measure proper acceleration. It works due to the physical inertia of a mass. Conceptually an accelerometer is composed by a mass connected to a damped spring. When the accelerometer is subjected to an acceleration the mass moves respect to the container, proportionally to the acceleration module. Measuring the displacement give the acceleration which the

devices is subjected to. The acceleration is then encoded in some way into an electrical signal. Accelerometers are used for example in navigation systems to get the relative position referring to a zero point, in our smartphone as input devices, to rotate the screen, to play some game, as pedometer and so on, or in some computers with magnetic hard drive as falling detectors to prevent hard drive issues setting the reading head in a safe position.

At this link you can find an example of using a 3 axis analog accelerometer, that give three voltages proportional to the accelerations, with a programmable board:
<https://www.arduino.cc/en/Tutorial/ADXL3xx>

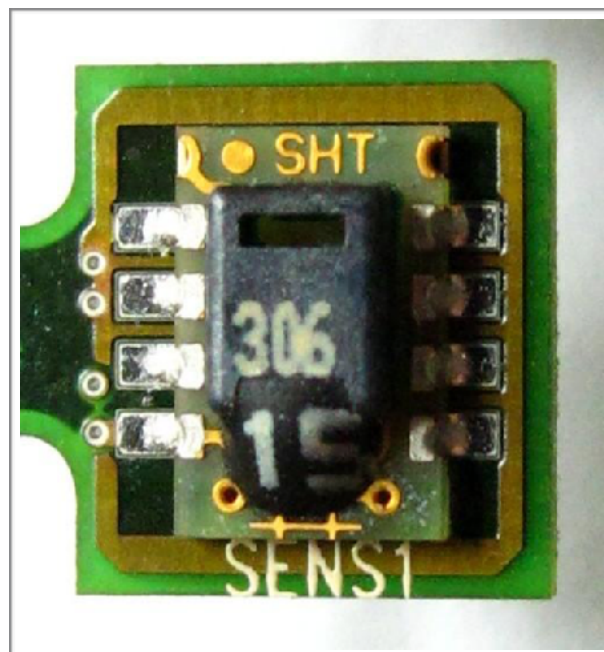


Humidity sensor

Humidity sensors measures humidity in a medium, like the air or the topsoil. They mostly works measuring the variation of a capacity and or resistance due to the variation of the amount of water in the medium. Humidity sensor are used for example in metereology, also in our meteorological portable station, and in automatic irrigation system that can open the water only when the soil is dry.

At this link you can find an example with an open source board and a very common humidity sensor (that can also measure temperature):

https://www.tutorialspoint.com/arduino/arduino_humidity_sensor.htm



Temperature sensor

A temperature sensor convert the temperature into an electrical signal. This is possible due to different physical phenomena, depending to the type of the sensor. For example the thermistor offer a variation of its resistance according to a variation of the temperature, and the thermocouple offer, for the thermoelectric effect, a voltage proportional to the difference of temperature.

Temperature sensors are used in meteorology, medical thermometer, computers CPUs etc.

Here you can see an example temperature reading using a programmable board:
<https://learn.adafruit.com/tmp36-temperature-sensor/using-a-temp-sensor>

Output: a short overview

An output is an element that can act on the real world using different ways of interaction. The most important output using movement, lights or sound to interact with users and with the environment.

LED

If you want to introduce LEDs in your class you can use a lot of different methods depending on the type of your school. In this manual we will generically show you all the electronic components you will need so that teachers from different disciplines can use them without any troubles.

Before we define a component we need to bind the creation of the component to its inventor's life. The inventor of the LED is Nicholas Holonyak Jr, an american with russian origins, who has developed the first LED back in 1962, it was able to emit only the red light, nowadays it's possible to buy LEDs of very different colours. For example thanks to Asaki, Amano and Nakamura blue LEDs have been created, this research was so important that they won a Nobel Prize.



A LED is an apparently very simple output which allows to emit light of different colours in a reliable way, it lasts for a long time and it's low cost. Today LEDs are used for many applications, not only in commercial but also in householding purposes.

The operating principle is based on electrons occupying some gaps in a semiconductor material and releasing energy, or the photons. The different frequencies, also different colours, emitted depend on the materials. The LED needs constant current to be powered, this is the reason why it goes always with a resistance that allows to control the electrical flow that reaches the diode.

More exercises:

Here's a list of some additional exercises (not treated in this manual) that will help your students to practice and improve their knowledge, it will be also useful to make teacher's evaluation easier:

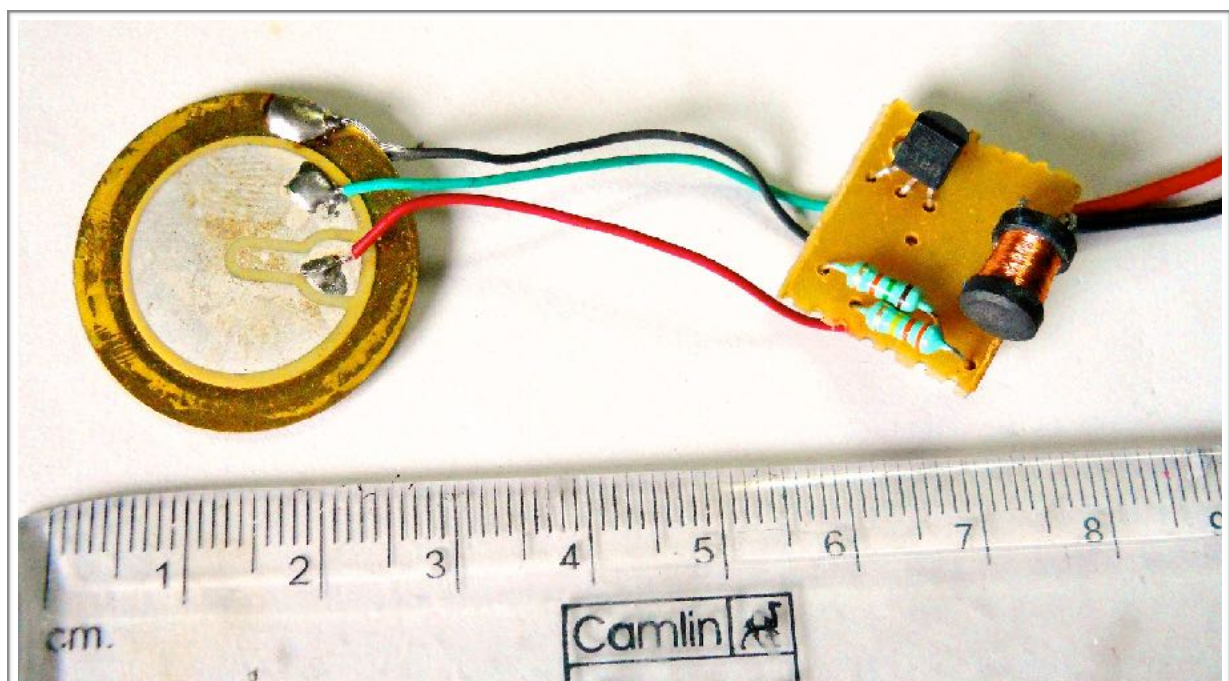
Change LED blinking time, from 1 to 3 seconds.

Change digital pin (not 0 and 1!) where the LED is connected and change the program accordingly.

Change the blinking time this way: LED on for 1.5 seconds- LED off for 2/3 seconds.

Buzzer

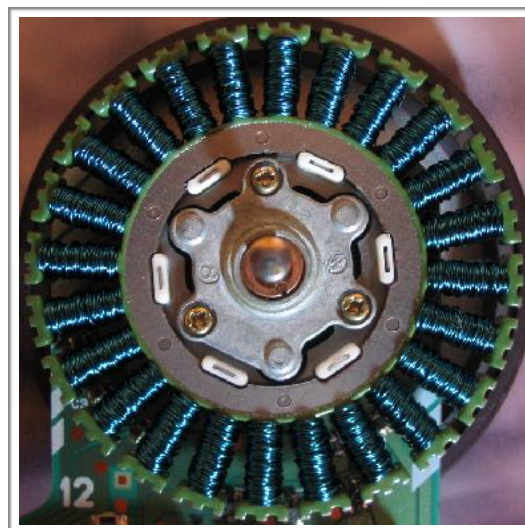
The first thing is that you can use Arduino to emit sounds with different tones and lengths. Therefore you can talk about music and students' favourite songs and musics. Furthermore it is possible to watch a part from S. Spielberg's "Close Encounters of the Third Kind" the one where humans communicate with aliens thanks 5 musical notes (the interpretation of the scientist that makes this scene start is Francois Truffaut's). Obviously you can make students make their own music. In this way you can involve students to use coding in a creative way. After making the



introduction on how to emit notes with the buzzer you can program the software to learn to play famous songs.

Motor

Motors allow to move for the first time the robot, the first movement of the robot will be very simple and silly, the robot will go forward infinitely careless about obstacles and the operating time. In the Arduino world The H-bridge is a transistor that allows us to control two DC motors simultaneously. Without the H-bridge it wouldn't be possible to make the robot move. The H-bridge in fact is a battery that manages the energy supply of the robots. Motors included in the most popular Arduino kit are DC motors without any sensor. That's why it's not possible to manage motors' rotation but only the current delivered to them.



“Every maker of video games knows something that the makers of curriculum don't seem to understand. You'll never see a video game being advertised as being easy. Kids who do not like school will tell you it's not because it's too hard. It's because it's--boring”

— Seymour Papert

Design Challenge

In this chapter we present some guided design challenge with different levels of difficulties and different hardware and software to use.

Every Design Challenge is thought as a challenge to solve with our classrooms, using some educational methodologies:

- 4 P: Play, Passion. Peer and Project
- PBL: Project Based Learning
- Flipped Classroom

Here some short definition of each methodologies that we will apply on our educational challenges:

4P: Project, Play, Passion and Peer

Mitchel Resnick introduce with the book Kindergarden: LifeLong learning the concept of creative learner.

A creative learner (student) needs to solve problem focusing in the process. At school obviously we can't solve problem in a drastic ways but we can learn one or more process that could allows to learn.

Projects:

A project is a process that can solve a series of problems that are composing the issue that we want to solve.

We don't need a concert project, we can also use theory to understand the world a find proposal on the improvement of one or more problem.

Peer:

Collaboration is fundamental in the learning process, teaching to our peers, share information and listen our colleagues, classmates allows us to improve our knowledge and our awareness of our friends and audience. Thank to this part we can developed empathy.

Passion:

It means Engagement. We can work with the positive feelings of our students when we are using technology,

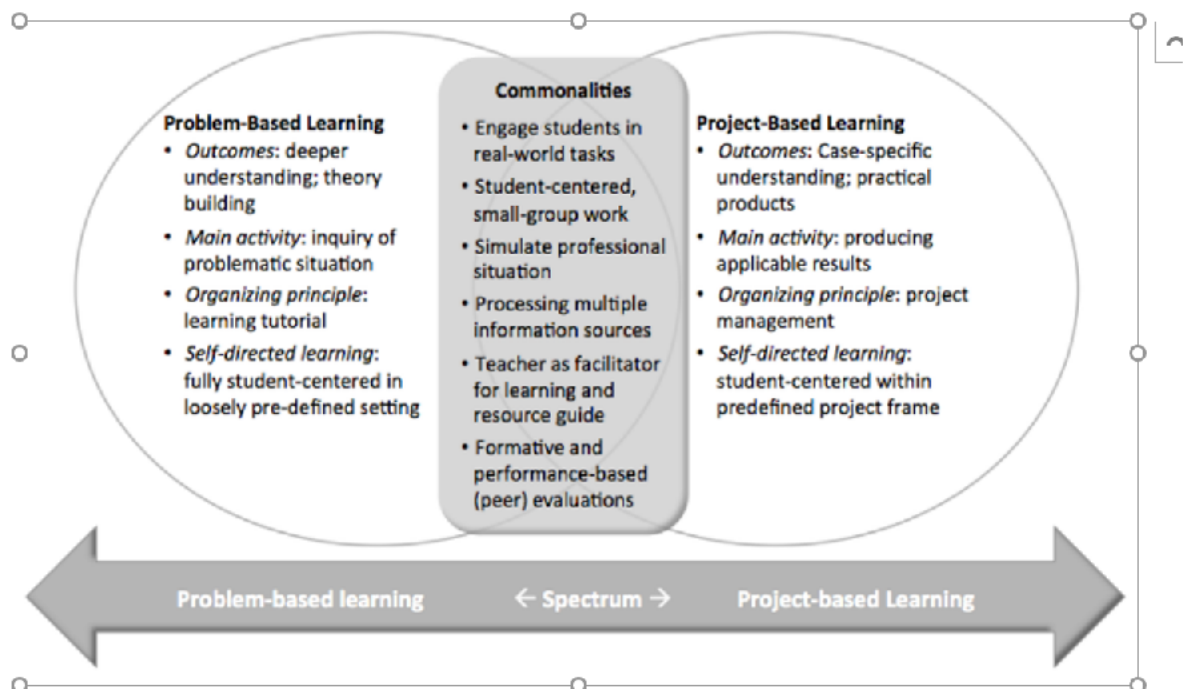
Play:

We need fun during our learning process. A simple environment, happy faces allows people to learn better and faster.

Project Based Learning

To encourage creative learners PBL is a methodology based on the study of a project that can involve all disciplines. This means plan real world scenario for the student where ideas can be connected with real problem solving. Thanks to PBL methodology students are more engaged and can enhance collaboration, soft skills, foster creativity and make learning fun!

From wikipedia we can discover an interesting image to discover better what is PBL:



Flipped Classroom

This methodology comes from blended learning methods and forecasts a change of the classroom, where students have to study different subjects and teach their classmate what they learn. In this process the teacher's role can change and become a sort of coach, planner of the learners and push them to discover with working group and presentation new things.

Design Challenge 1: Magic Wand

Robotics kit: Lego Wedo 2 or Arduino or Microbit (in this case is developed for Lego Wedo 2)

Software: Scratch 3.0 and Lego Digital Designer

Age: from 10 to 15 yo

Introduction

We have to assemble a wand with Lego WeDo 2 pieces, it is very simple and does not require manuals. For this reason, from an educational point of view, the lesson can be organized starting from the project of the wand both on sheets and using the free software Lego Digital Designer.

In this phase the students will have to:

1. design the wand on paper or computer (via LDD)
2. make the wand designed with Lego WeDo with the only care to insert the TILT sensor into the wand handle.

The aim of the exercise is to create attention and involvement on the part of the students using common narrative techniques (stories on magicians). Thanks to use of the wand will be easy to introduce the use of the tilt sensor and understand the use of the "If" on Scratch.

Storytelling

The exercise lends itself to working a lot on the narrative part, children can become part of a story in which there are protagonists magicians. They can prepare magician's hats and put them on for children during lessons. You can work on fairy tales dedicated to magicians using the construction and the next programming spells as a phase of the overall narrative.

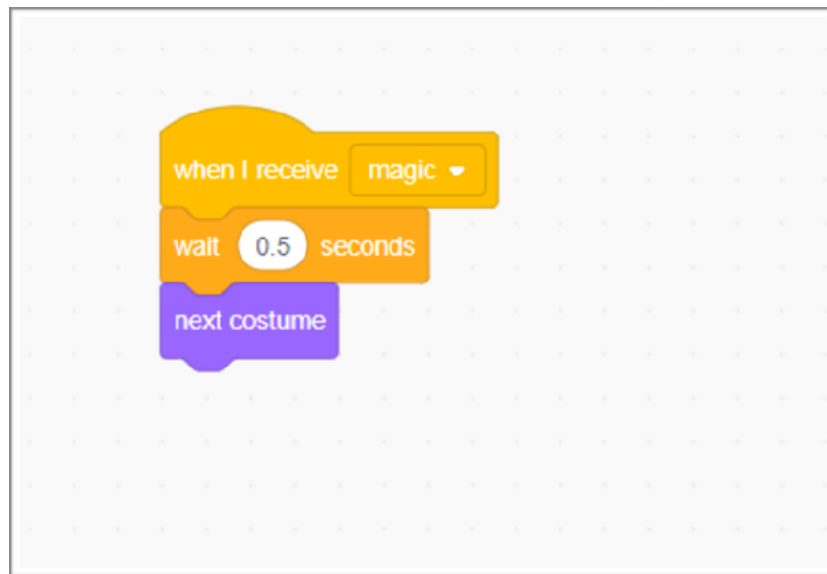
First Program

The first program is used only to manage the wand and to change the costume of a sprite inserted in Scratch when the wand is moved.

For example, changing the position of the wand from vertical to horizontal will change the shape of the sprite.

No special pre-requisites are required on the programming language to run this program. In this simple exercise students can discover in computer science 2 important block:



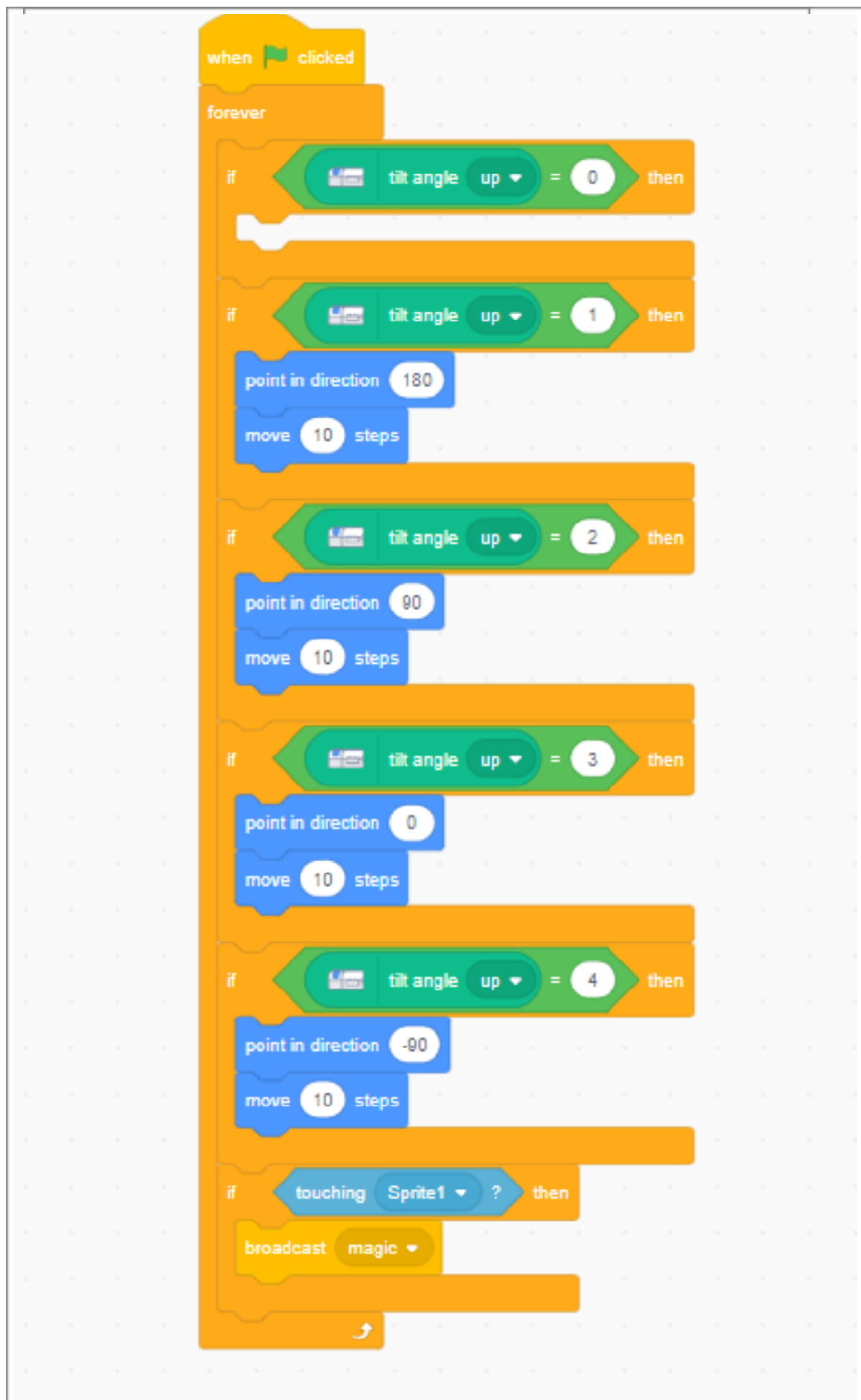


This block allow to create a loop repeat until infinite loop. In this case it mean that we control every time where is the position of the wand.

The second important block is IF-THEN thank to that we can compare the position of the sensor and if the angle is up we can change the image in the screen.

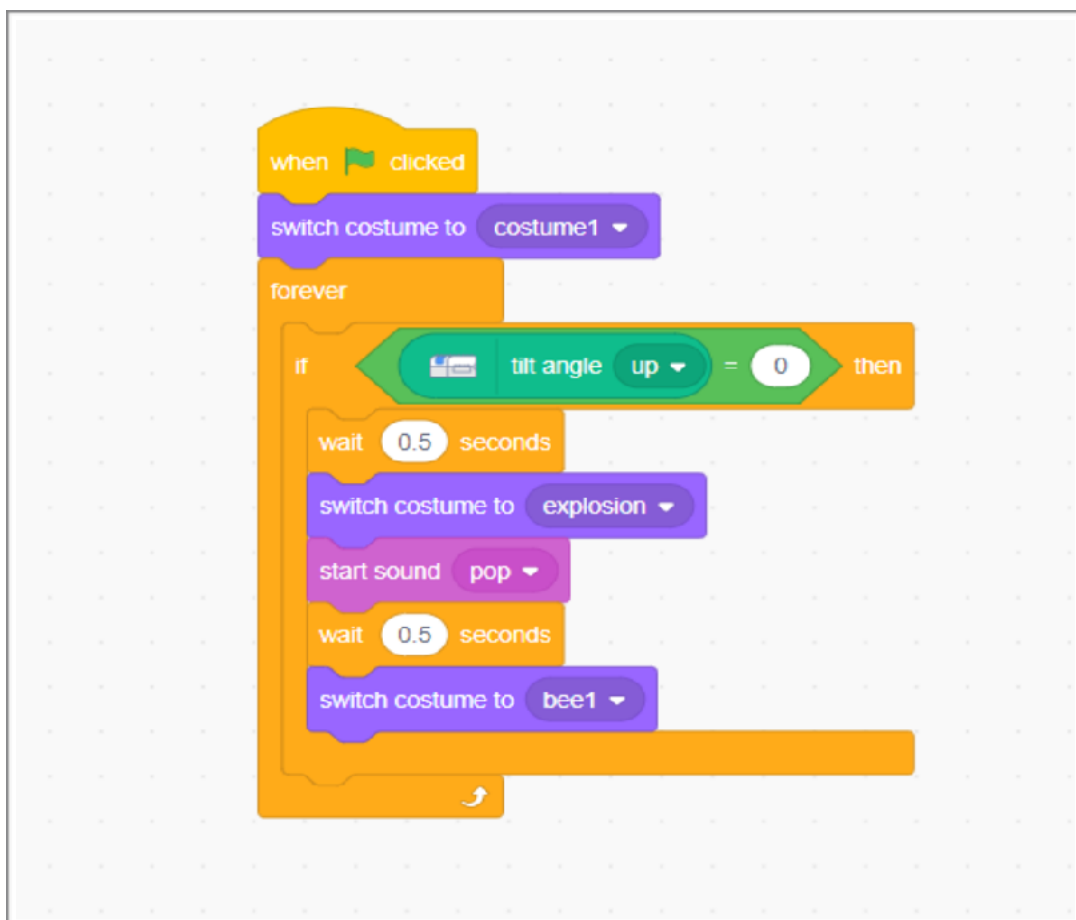


Thanks to this simple program, students will be able to understand the potential of the "IF" command and construct a simple flowchart that includes an or more options based on the TILT sensor status of the Lego WeDo.



Program 2

The second program does not envisage any extension of the teaching horizon, but only the consolidation of the newly learned concepts. The second exercise provides that at the time of the execution of the magic there is a sound and a graphic effect that heralds the magic. From the didactic point of view there are no new concepts, so teachers can use this exercise to understand if the students understood the potential of the software.



Moreover this part of the program allows a strong personalization by the student (not only the sound but the design of the magic effect). There personalization allows a greater involvement of the students and a greater and more immediate learning.

Program 3

On the first two exercises you can make numerous variations on the subject, of equal difficulty between them. To increase the level of programming you can change to this way the request to the student: "create a magic wand that becomes a controller of a star on the video. Everything that is touched by the wand it is transformed ".

This request involves the creation of new algorithms:

1. movement of a sprite according to the position of the TILT sensor (4 states identified);
2. linking one programming flow to another (present in another sprite) using the "Broadcast" command.

Design Challenge 2: Fish and tilt sensor

The aim is to put into play all the knowledge acquired up to now in a complex program, which involves the interaction between many sprites. The variable used (already present in Scratch) is called "dimension".

This part is dedicated in particular to the secondary school of first grade and the first year of secondary school. The structure of the pantry is slightly different from the previous ones to accompany the teachers in Scratch's programming: there will be no exercises graduated, but only one described step-by-step.

Purpose

The aim is to create a game in which a predator fish (only digital or even "real"), commanded through the Tilt sensor, can eat only the smallest fish. At each fish eaten the protagonist, the predatory fish, grows in size. The enemy is a much larger fish, a super-predator, than the protagonist fish it will have to be avoided until it has become big enough that it will not take more risks. Only then will we have reached the end of the game.

Storytelling

To build this videogame you have to introduce some elements related to the plot of a story. We need to introduce the figure of the protagonist, the purpose of the game (eating other fish) and the figure of the antagonist, the enemy, which hinders the achievement of the goal.

This can be a first step to the actual introduction of a story in which all the actors are clearly defined: protagonist, helper of the protagonist, antagonist, helper of the antagonist, magical object, aim to be reached.

The videogame

To be able to realize the videogame, the programming of each step-by-step element and an image of the finished game is presented.

Script

Background (Stage)

As in the pantry 1, you can import a background and edit it with the drawing editor, or if it is can create a new one, as was done for it cloud sprites. In this exercise it is not necessary to associate no background action, but it is possible to predict the alternation of different backgrounds; in this case they must be displayed at appropriate times, planning the appearance.

Sprite Protagonist - Movement

To control the sprite of the protagonist fish just insert the script in the main script created in the exercise program 3.

Sprite Protagonist - Variable initialization

"dimension"

This concept underlies any code provides for the initialization of the variables of the program. In this case the variable that we will use is already defined by Scratch (not we have to create it ourselves) and it's called dimension. Initialization is done using the command

"Port size to%" in the folder

Appearance (top left). We bring variability

"Size" at 30%.

Sprite Small fish - movement and interaction

In the videogame there will be many small fish, all using the same script. just

make a single sprite "small fish" and then duplicate it as many times as necessary. All sprites in the video game must be activated through the same button, in this case the "Space" (these are actions that are activated "at the same time").

When programming a videogame you have to think that the state "show" or "hide" is from initialize, especially in the event that during the game you think to also use the command "Hide".

To start we will have to consider showing the sprite and set its initial size to 20%.

We then proceed to program the movement and interaction. The movement of the fish takes place along a line but you must have the precaution of entering the command "Bounces when you touch the edge", to guarantee the continuity of movement. If at the time of bounce the sprite turns it upside down is enough select the "turn only right" button and a left "(as shown in the picture is the button square in the middle with horizontal arrows).

Sprite - Protagonist

To program correctly you must consider that, at the time when the sprite of the protagonist is touched by a fish, must the evaluation should be made if the fish is more big or smaller. If the protagonist meets a smaller fish (<30 in our settings ...), which yes realizes with the command "when I receive ... size xx ", it will be enough to suggest to the program increase the size of the protagonist of one percentage of our choice (in the example 10).

Sprite - Big Fish

As in the case of small fish will have to be shown at the beginning. Unlike small fishes yes will make the big fish appear in one position random (x, y), managed by the combination of numbers random that will be inserted with the appropriate script

(as shown). Random will also be the direction of movement of the fish. The initial size of the big fish comes set to 70%. To guarantee a varied movement throughout the space available, you can set the fish to bounce back to each edge and having a random speed (the number of steps is randomly generated from 1 to 15). As for small fish, if the big touches the protagonist must send a nominated message as its size (in this case 70).

Sprite Protagonist - receiving message from great fish

When the protagonist receives the message from Big fish must understand if it will be "eaten" or if you can face the opponent, condition that will be fulfilled when his size will be at least 60%. If the condition is verified, the video game ends with the extreme magnification of the protagonist. In case the protagonist is smaller than the Fish superpredatore, the game will end with a written represented by a sprite that will be activated at reception of the "game over" message.

Sprite Game Over - Script

Game Over is a sprite designed by the user. The word Game Over is activated only when it receives the Game Over message, while at the beginning of the program is hiding.

Sprite Big Fish - Victory

The protagonist in the case of victory sends a "won" message to all sprites. When the message reaches the sprite of the Great Fish this must disappear. If desired, it can be added a sprite "You have won". In the proposed program the Predator fish becomes huge and continues to to swim.

Recap

You can find the exercise here: [XXX LINK](#)

The realization of this software is more complex so in order not to weigh down the reading we have not inserted didactic notations in the text as in the previous handouts. The teacher in conducting a laboratory dedicated to the realization of this game will have to clarify some key concepts.

The "dimension" variable

Scratch allows to create new variables but some are already inserted and defined. "Dimension" is the classic example of a variable local, which can only apply to the sprite for which it is used, unlike other variables that can be used by more sprites.

The use of this variable does not allow for example to recall it in the scripts of other sprites.

Thanks to "dimension" the concept of variable can be explained. In this game it is clear that the change in the variable dimension decrease the success or not of the

game. Very often when the variables are introduced at school, we risk posthumous examples, created ad hoc e not very functional or too complex examples. The use in this video game, however, provides a soft approach to the question variables.

Random numbers

To generate fish that move at random speeds you need to use the Scratch random number generator, this allows us to make students understand how these "generators" are used in the world of computing.

Send - Receive

In this program, communication between sprite is essential. This allows us to introduce parallel and programming flows to make complex preparatory flowcharts.

Design Challenge 3: Oracle

Software: Scratch 3

This Challenge is link to the world of vocal assistant that is an impotant issue on the market and all the most important companies are producing their vocal assistant. To start the lesson we can inspire our students starting with a sci-fi movie, 2001 Space Odissey and after with an overview of some important vocal assistant In Kubric's movie 2001 Space odyssey we can discover HAL2001 an assistant that can interact with asotrnavts and control the space ship. After that we can also quote the world of Star Trek, where all crew can speak with the “ship” asking help, suggestions, and service.

We can connect with the real world presenting all (of almost all) real products on the market, here some examples:

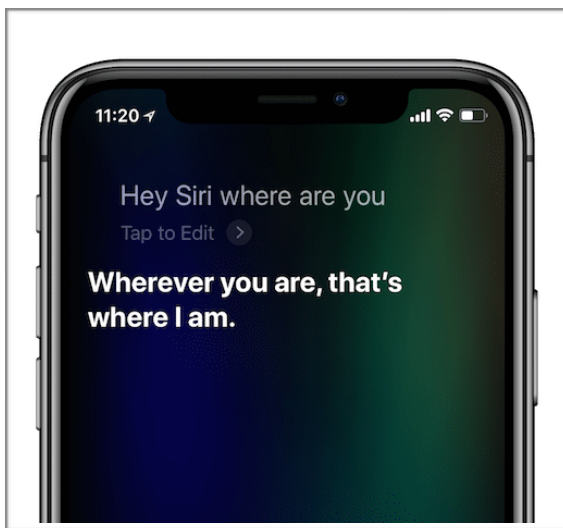
Alexa (Amazon)



Google Home (Google)



Siri



All these vocal assistants can answer your questions checking info on internet and using the most important apps like online music platform, cloud services, online market.

From an ethical point of view we can quote with students this case study link to the use of the voice of a famous singer (as happens in a Black mirror). The use of a famous singer is creating some cases of a wrong use of this vocal assistant that is substituting the real friends of teenagers.

From an ethical point of view it is important to quote this case study and use with students to introduce an ethical use of technology.

The introduction of vocal assistant can be used also to introduce AI concepts.

Vocal assistant and Internet of things

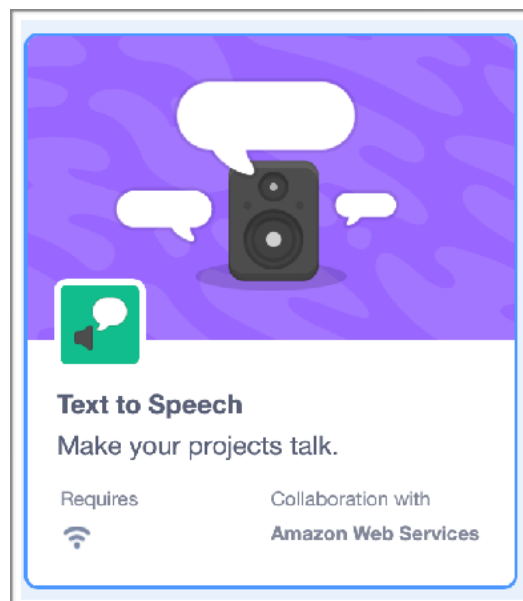
We present the use of vocal assistant alone but in a near future all this assistant could be the principal interface of all things connected to internet in our daily life.

After this introduction we can discover how to program a vocal assistant with Scratch 3:

In Scratch it isn't possible to use voice recognition but we can create a program where the interaction will be with voice from computer and text from the human user.

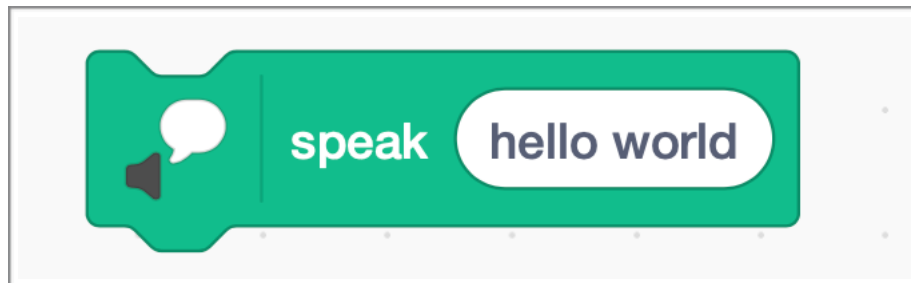
To program a vocal assistant, a sort of oracle, we can use the command "Text to speech"

To use this feature, Scratch called it Extension, we have to click in "more extensions" and we can select this image:



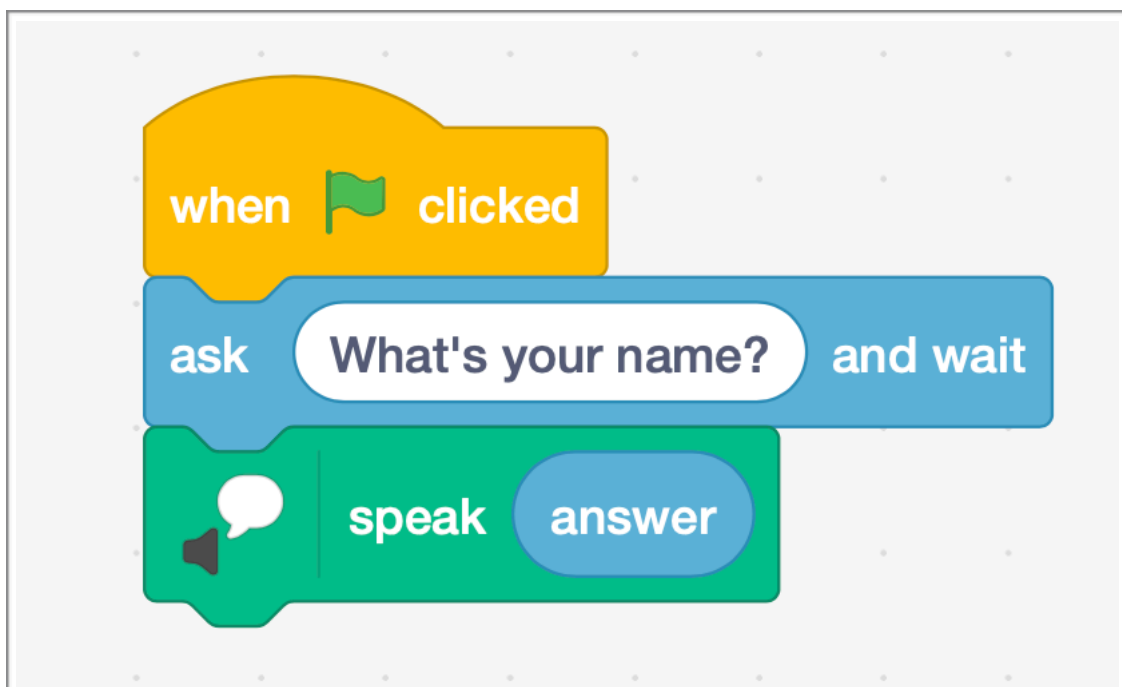
What is text to speech?

This block allow the computer to read hat we write in a text format, so the interaction it will be interesting.



We can use these blocks

Thanks to this program we can ask the name and write it by keyboard. Using Text to speech the program is able to read the name. We can improve the program asking some things and receiving different answers.



Focus on micro:bit



Scratch with micro:bit

Micro:bit is a pocket-size microcomputer designed to help kids learn to code and create with technology. It has many features including an LED display, buttons, and a motion sensor. It can be connected and programmed with Scratch, MakeCode and Python.

The BBC micro:bit contains an accelerometer which can detect if it's shaken or which way the micro:bit is being held. As well as detecting if it's upside down, accelerometers can detect some of the forces that are acting on it. The micro:bit has a built-in temperature sensor that can detect the current temperature of the device, in degrees and Celsius. The full list of features is presented in the next table:

Feature	Description
2 buttons	Programmable action push buttons
25 LED lights	Can be individually programmed to show shapes, text or numbers
USB connector	Connect to a computer for power or to load programs onto the micro:bit
Accelerometer	Senses if the micro:bit is being moved, tilted, shaken, or in free-fall, and at what acceleration
Compass	Detects which direction the micro:bit is facing
Processor	Where the program is executed
Radio	Communicate with other micro:bits for multiplayer games
Bluetooth antenna	Wirelessly sends and receives signals to Bluetooth enabled PCs, Smartphones, or Tablets
Reset button	Restarts the micro:bit
Battery socket	Power the micro:bit using batteries
Temperature sensor	Detects the current temperature of the micro:bit in degrees Celsius
Light sensor	The LEDs on the micro:bit can also act as a light sensor to detect ambient light
Edge Connector	25 external connectors, called Pins, on the edge of the micro:bit allow you to connect to other electronics hardware, including LEDs, motors, and other sensors. These can behave as inputs or outputs.

For the moment, Scratch 3.0 doesn't have code blocks to use the compass, thermometer, or light sensor. Also, only 3 of the 25 pins are used.

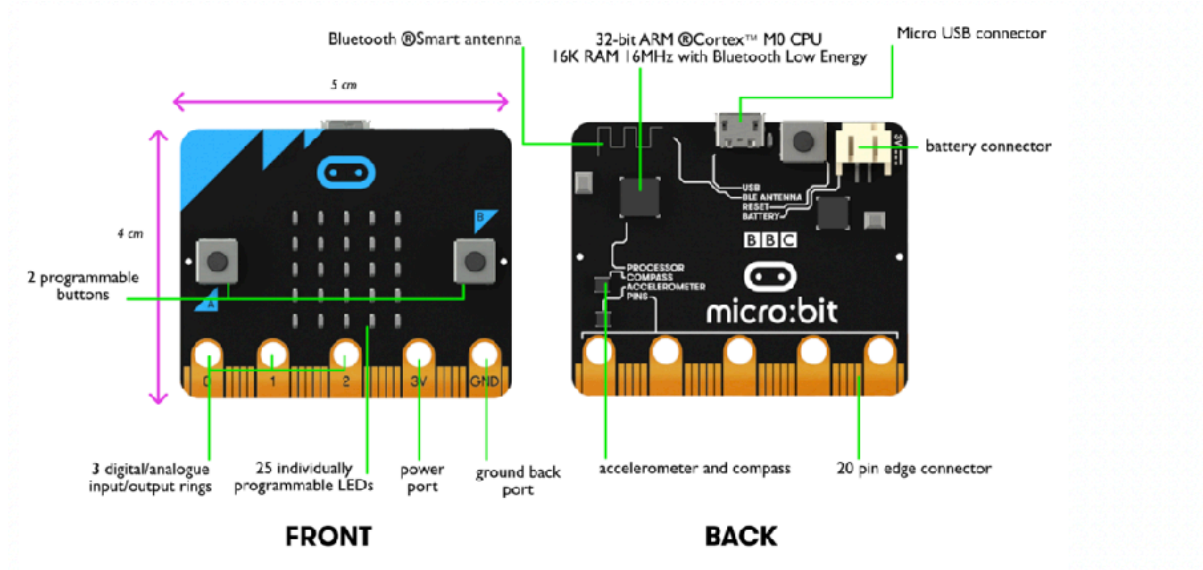


Image source: <https://www.pakronics.com.au/pages/microbit-in-australia>

How to use it

Micro:bit can be programmed on both desktop (Macs, PCs, Chromebooks, Linux, including Raspberry Pi) and mobile. In order to use micro:bit with Scratch next steps must be followed (for Windows).

Step 1: Install Scratch Link from Microsoft store or use the links from <https://scratch.mit.edu/microbit> page

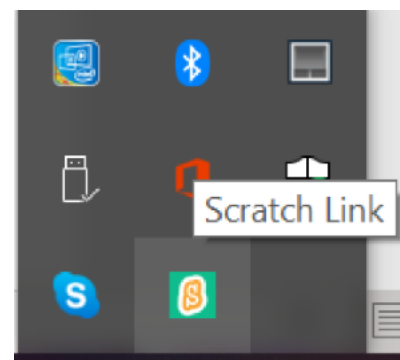
Step 2: Start Scratch Link and make sure it is running. It should appear in your toolbar.

Step 3: Connect the micro:bit to your computer with a USB cable. The micro:bit will show up on the computer as a drive called 'MICROBIT'.

Step 4: Download from <https://scratch.mit.edu/microbit> the Scratch micro:bit HEX file

Step 5: Unzip the archive and drag and drop the HEX file onto your micro:bit

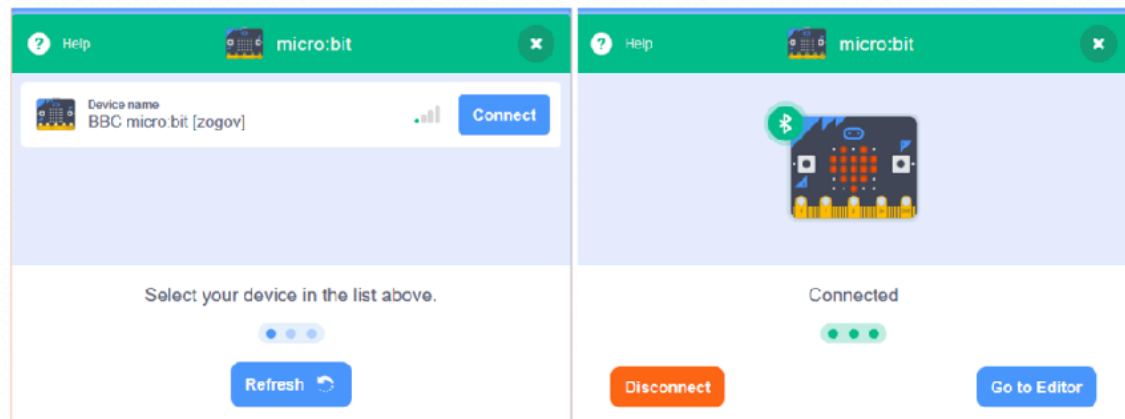
Step 6: Power your micro:bit with USB or a battery pack.



Step 7: Use Scratch Editor

Step 8: Add the micro:bit extension

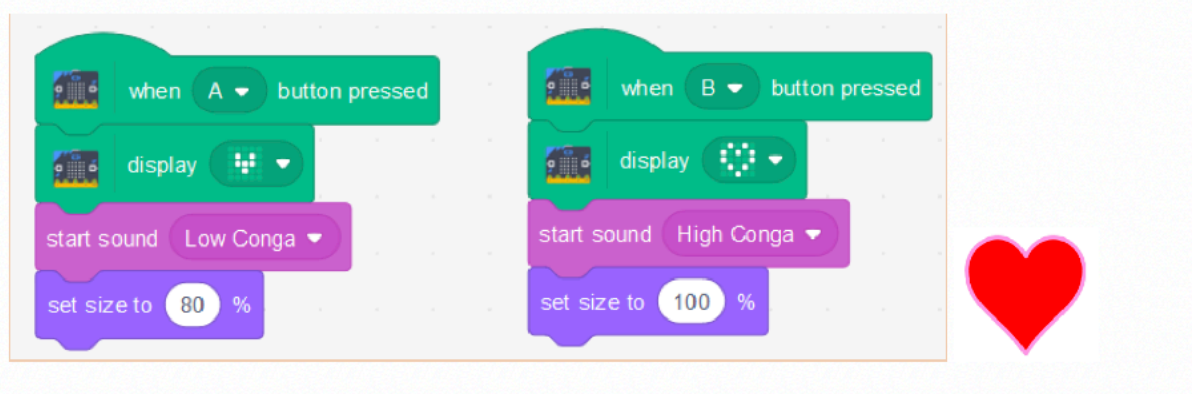
Step 9: Press Connect and Go to Editor



Design Challenge

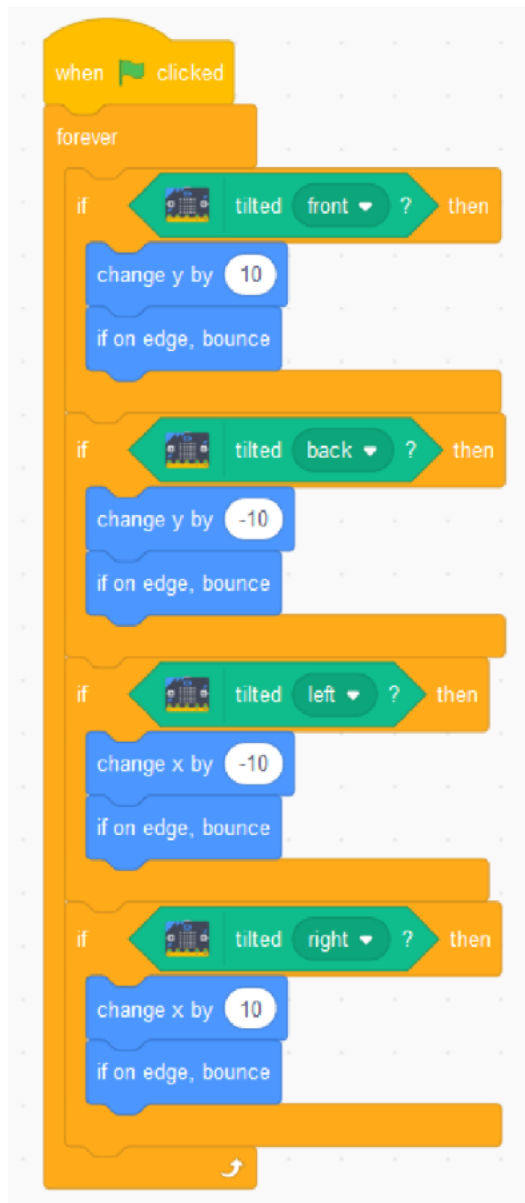
Design Challenge 1: A Heart Beat

The application (from <https://scratch.mit.edu/microbit>) is controlled by the two buttons of micro:bit. Depending on the button pressed it is displayed on the micro:bit the two hearts (one smaller and one bigger) and, in the same time, in Scratch a heart is changing sizes and two different sounds are played.



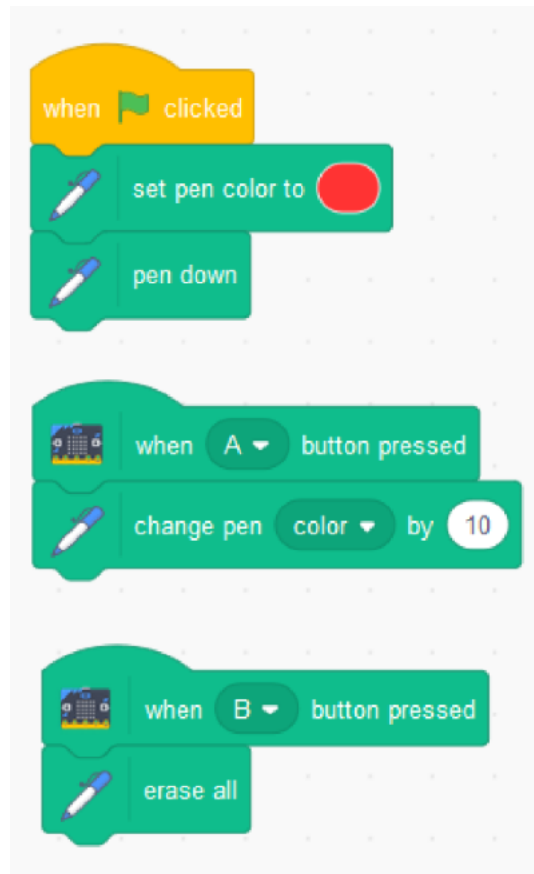
Design Challenge 2: Ball

By changing the inclination of micro:bit a ball may be moved on the screen.

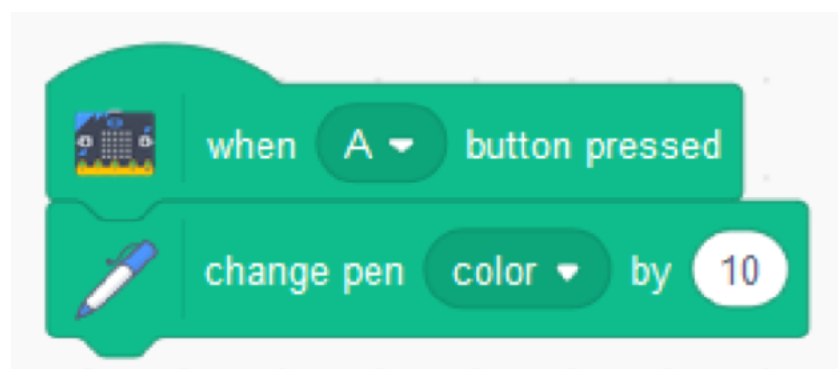


Design Challenge 3: A Pen

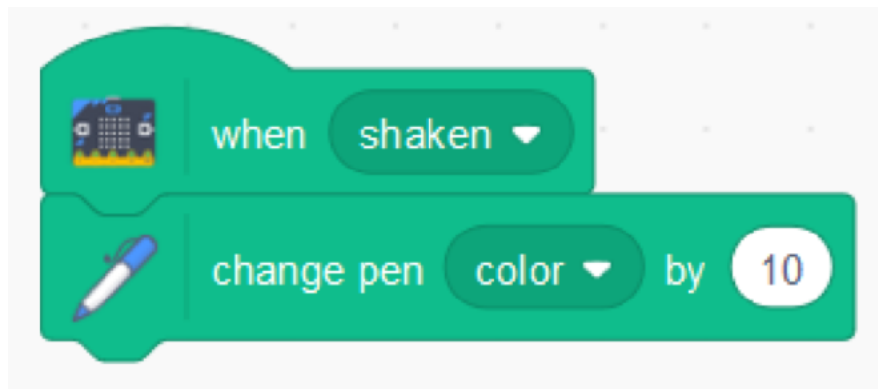
Add the pen extension to Scratch. To the previous application, add the next scripts. Test the result.



If you replace



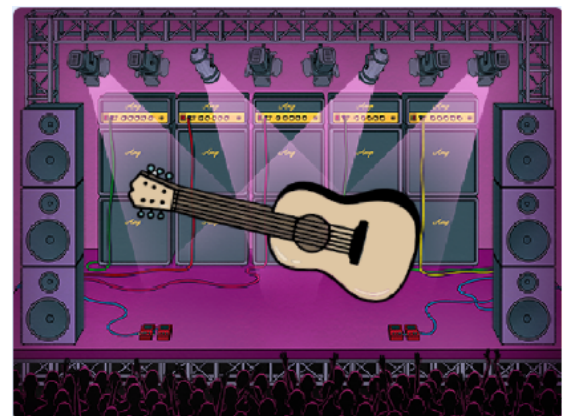
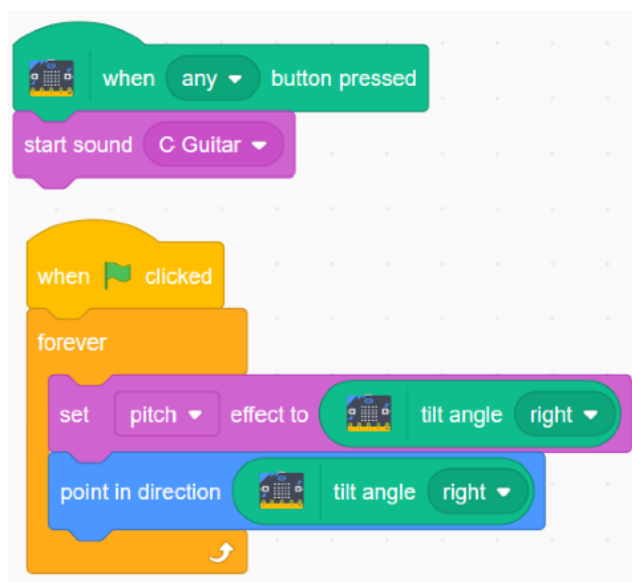
With



Then the color will be changed when the micro:bit will be shaken.

Design Challenge 4: Play Guitar

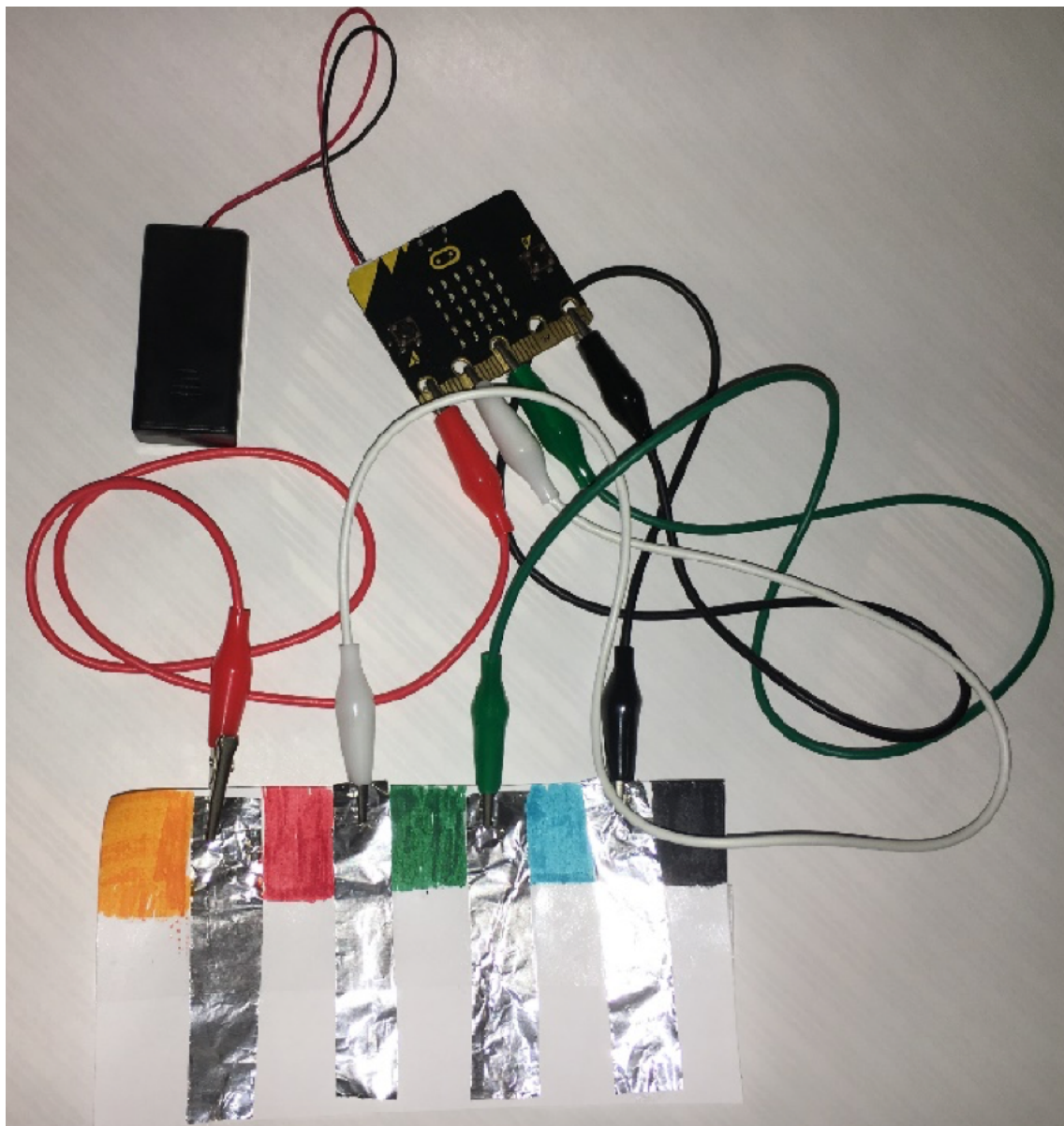
The application play a sound when any micro:bit button is pressed with a pitch effect depending on the inclination of micro:bit.



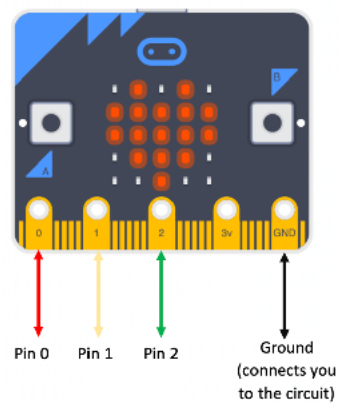
Design Challenge 5: Play a DIY Piano

In order to build the next application (inspired by an example from <https://microbit.org/scratch/>) you will need the micro:bit chip, 4 cables with alligator clips, tinfoil, scissor, glue and paper.

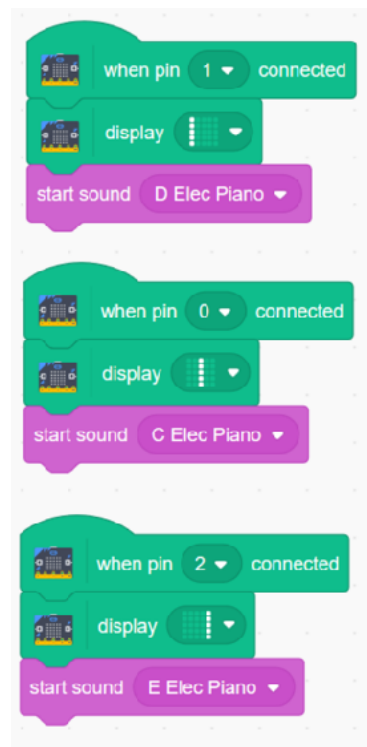
Build a piano like the one in the next picture. Connect each pin from 0 to 2 and the ground to one tinfoil band.



The scheme of this piano is in the following picture:



Write the scripts:



Test the application by touching the piano (tinfol bands) (one finger should touch the ground connected tinfoil band in order to close the circuit).

More Challenges to do

1. Build a “rock, paper and scissor” game. The application should display randomly a rock, a paper or a scissor when you shake the micro:bit.
2. Create your own wearable device with an interface in Scratch. Design a case/ support for your device and print it with the 3D printer.

Resources

<https://microbit.org/scratch/>

<https://llk.github.io/microbit-extension/iste18/>

<https://make.techwillsaveus.com/>

<http://libraryadventuring.blogspot.com/2018/10/coding-and-making-with-bbc-microbit.html>

<http://blog.sparkfuneducation.com/five-wearable-projects-with-microbit>

<https://scratch.mit.edu/discuss/youtube/44Xo76Bbqil/>

References

<https://microbit.org/scratch/>

https://diyodemag.com/education/kids_coding_scratch_30_meets_micro_bit

MakeCode for micro:bit

Microsoft MakeCode is a web-based environment for learning to code with physical computing devices such as the micro:bit. MakeCode is free and works across all platforms and browsers.

MakeCode is available as online editors for:

- micro:bit - <https://makecode.microbit.org/>
- Circuit Playground Express - <https://makecode.adafruit.com/>
- Minecraft - <https://minecraft.makecode.com/>
- LEGO® MINDSTORMS® Education EV3 - <https://makecode.mindstorms.com/>
- Arcade - <https://arcade.makecode.com/>
- Chibi Chip - <https://makecode.chibitronics.com/>



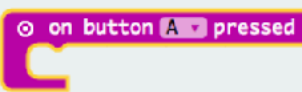





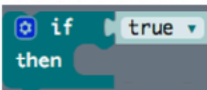



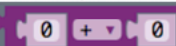
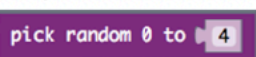
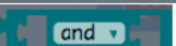

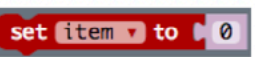
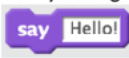
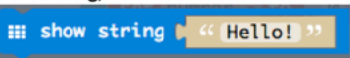


and as applications for:

- micro:bit
- Adafruit
- Cue by Wonder Workshop

The MakeCode for micro:bit application has a few extra features over the online editor. With the desktop application the micro:bit may be programmed directly over USB, without needing to drag-and-drop the file onto the micro:bit drive and directly read serial data from micro:bit.

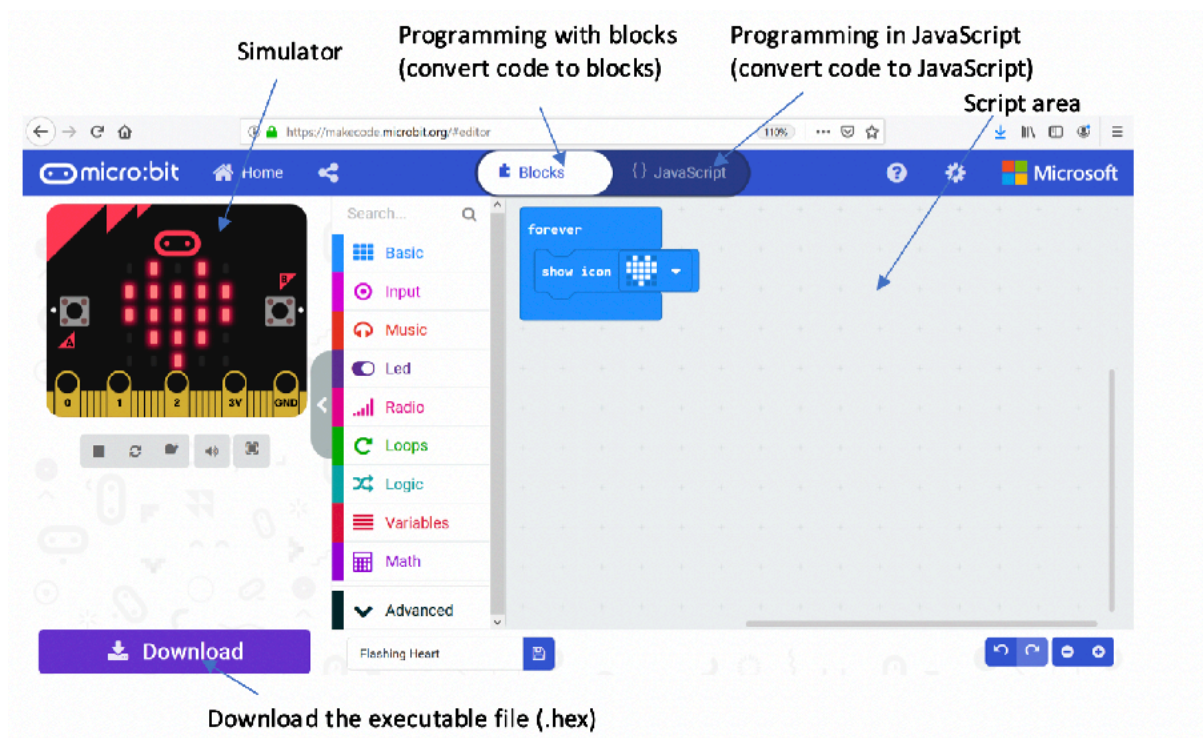
Scratch vs MakeCode for micro:bit

On micro:bit support page (<https://support.microbit.org/support/solutions/articles/19000080171-moving-between-scratch-and-makecode>) it is published a comparison table, listed below, between Scratch and Makecode for micro:bit, in order to emphasis the subtle differences between several important blocks. The table might be useful for the user of one of the two online editors, passionate about micro:bit, to start to use the other one.

Scratch	Makecode
Events wait for a user action, like clicking the green flag in scratch or pressing a button on the keyboard  	Input waits for a user input like pressing the A button or shaking the micro:bit 
Control is about the flow of your program tasks. In scratch you can add a forever , repeat or if block to an event to trigger it.	Makecode breaks controls into programming concepts. Forever is it's own loop (it's triggered as soon as the micro:bit is powered on). 
  	Repeats are found in the Loops menu of JavaScript Blocks  And if blocks are found in the Logic menu 
Operators let you do arithmetic and make comparisons   	Arithmetic and random number pickers can be found in Math , whereas a comparison between something and something else is found in Logic   
Data lets you define variables that might change within your program. Here we have made a variable called item 	We can define these in the Variables menu of makecode 
Looks let you display actions on the screen, which can be done by adding a say block. The word in the box is called a string 	To show a word on the micro:bit display we can use show string , found in the Basic menu 
Blocks that let you edit them have white backgrounds that you can type in for example operators have circular input areas 	Blocks that you can edit usually look like jigsaw pieces and may already have an example in them, for example Math sums have a '0' in the block 

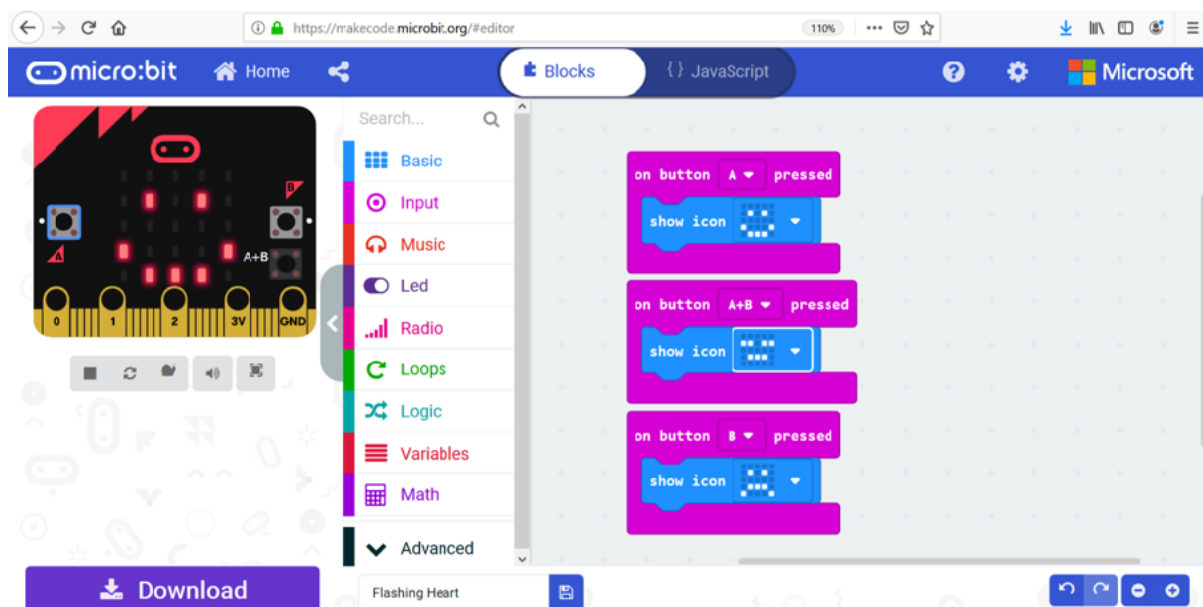
One big advantage of Scratch, comparing to online MakeCode, is that allows OTA (Over-the-Air) programming by Bluetooth communication. The big disadvantage is that Scratch provides a basic set of blocks for micro:bit, only 10. For this moment, the more rich set of micro:bit blocks in MakeCode makes it preferable.

Interface



Design Challenge 1: Faces

This application shows a smiley face when button A is pressed, a sad face when button B is pressed and an asleep face when both A and B buttons are pressed.



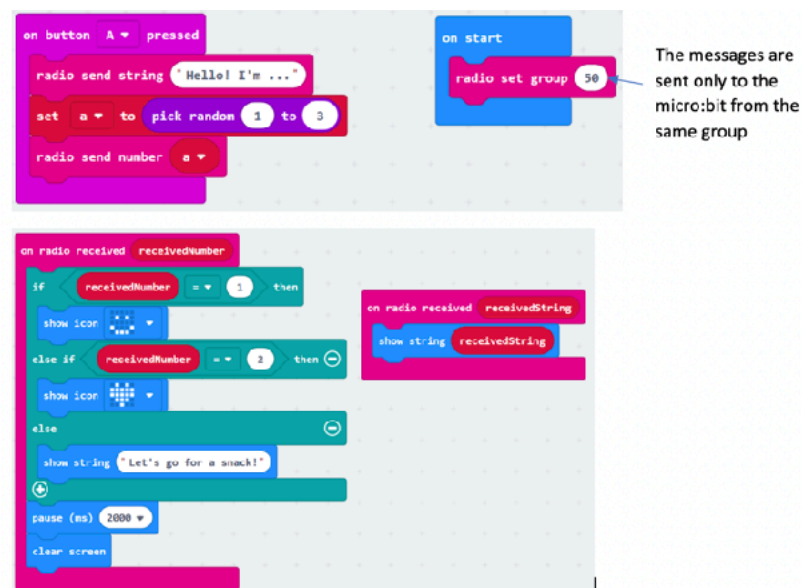
In the simulation area there are buttons A, B and A+B which can be used to simulate what happens when the micro:bit buttons are pressed.

In order to put the application on micro:bit chip you have to download the script (as a hex. file). Connect the micro:bit to your computer with a USB cable. Locate the downloaded .hex file and drag it to the MICROBIT drive.

This application is inspired from the tutorials from <https://makecode.microbit.org/projects/>. Try other tutorials!

Design Challenge 2: Chat

The next application let two or more micro:bit chips to communicate one to each other by radio connection. The *receivedString* and *receivedNumber* variables are dragged out from the *on radio received* block. The application must be uploaded on each micro:bit.



More examples with radio connection can be found at: <https://makecode.microbit.org/projects/radio-games> and <https://www.instructables.com/id/Radio-Signals-on-Microbit/>.

Design Challenge 3: Music and light

The next application allows to make music by varying the light intensity on the micro:bit light sensors.

Required materials:

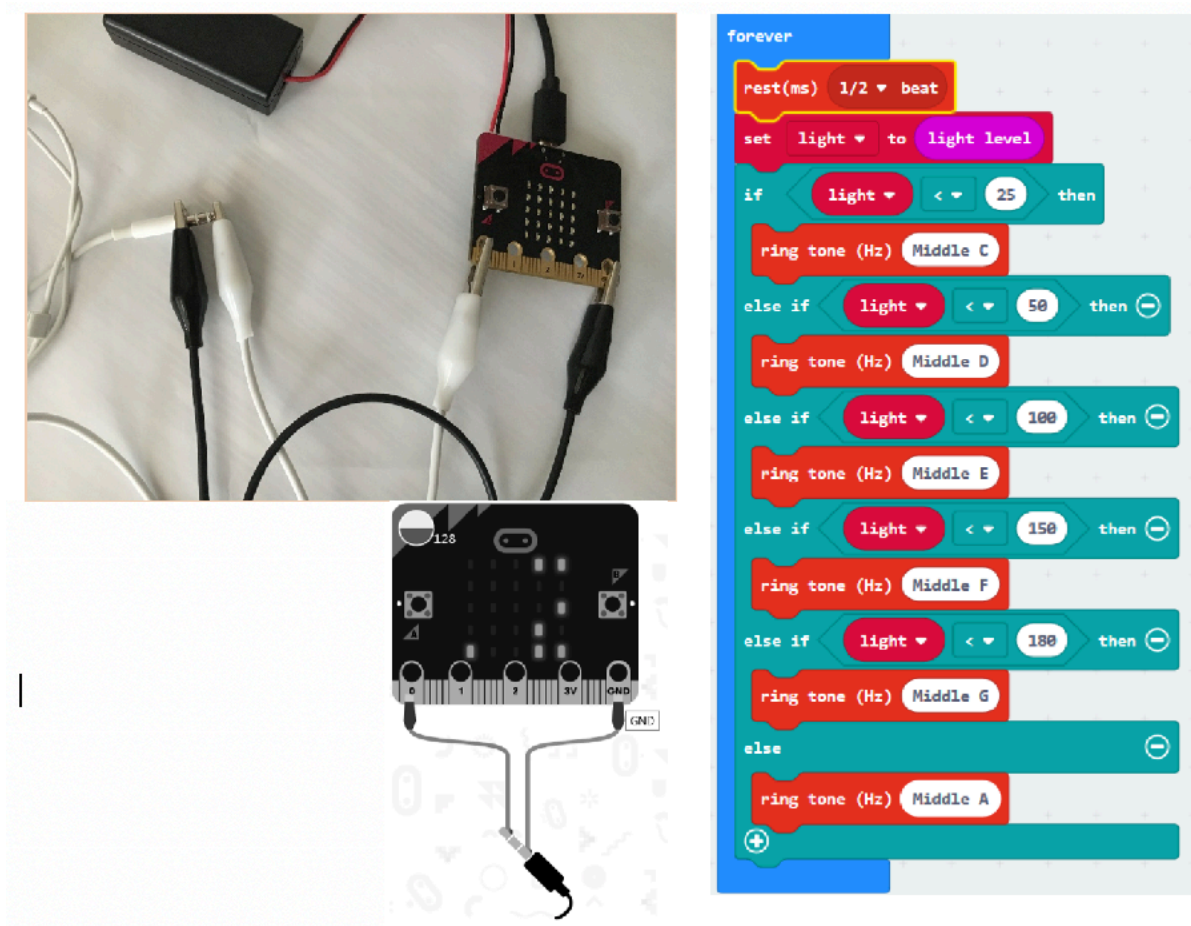
1x Micro:bit

1x Headphones

2 cables with crocodile clips

1x USB cable

For the beginning, you will have to connect the headphones to the micro:bit like in the next pictures:



Define a new variable called *light* and write the code. Copy the .hex file to the micro:bit drive. The light level is a value between 0 (dark) and 255 (bright). The light is measured by using various LEDs from the micro:bit screen.

Design Challenge 4: Music and... fruits and vegetables

This is a fun application which uses fruits and vegetables to close the circuit and make music.

Required materials:

1x micro:bit

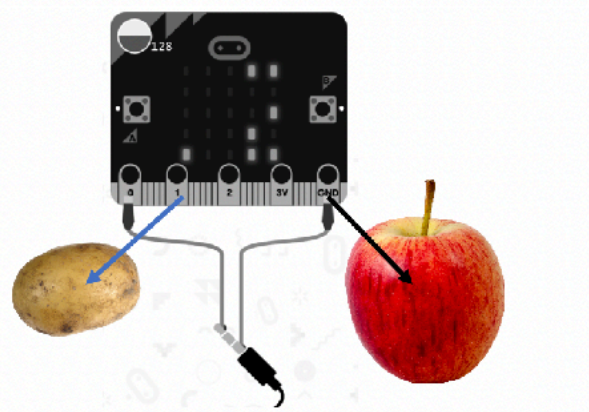
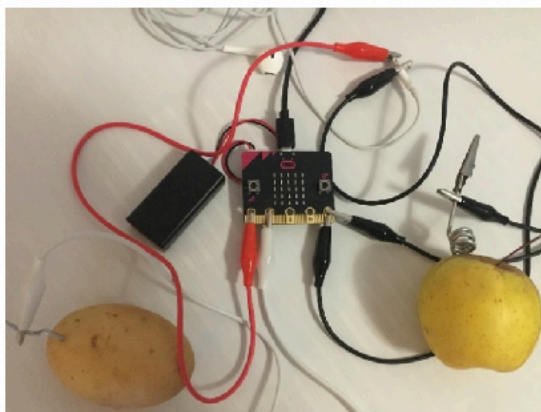
1x headphones

4x cables with crocodile clips

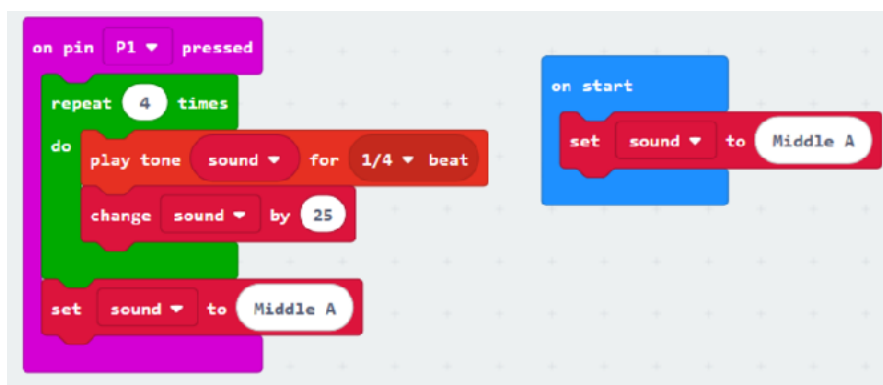
1x USB cable

2x fruits or vegetables (potatoes, banana, oranges, apples, etc...)

Make the following connections:



Create a variable named sound to store a musical note. Write the next code.



Save the .hex file and copy it to micro:bit drive. Create music by holding the fruit/vegetable connected to the ground (the apple in our example) and touching the other fruit/vegetable (connected to pin 1).

You may connect another fruit/vegetable to pin 2 to create other sounds. In this case you may duplicate the code for pin 1, select pin 2 and change, for example the value 25 with -25.

Design Challenge 5: Servomotor

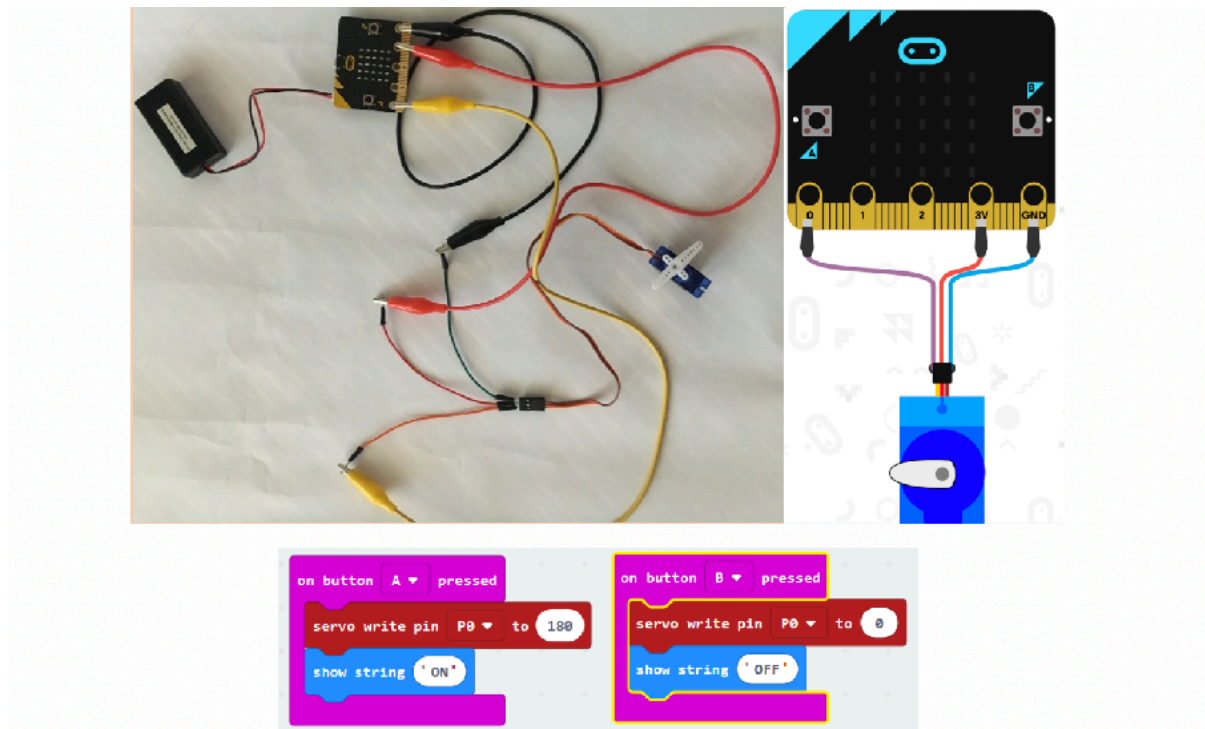
This application shows how to connect a servomotor to a micro:bit.

Required materials:

- 1x micro:bit
- 3x cables with crocodile clips
- 1x USB cable
- 3x male to male wires
- 1x servomotor TowerPro SG90

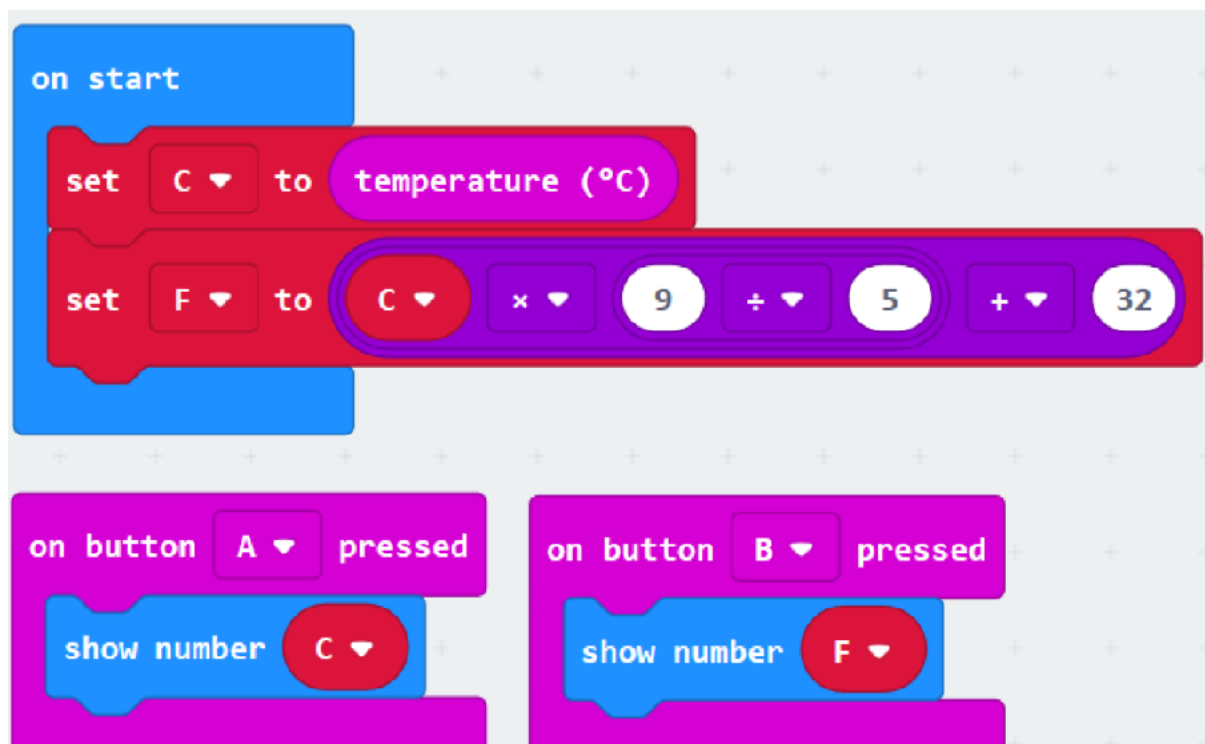
The servomotor must be connected as follows:

micro:bit	servomotor
GND	Brown wire
3V	Red wire
P0	Orange wire



Design Challenge 6: Thermometer

This application uses the build in temperature sensor to display the temperature in Celsius degrees, when button A is pressed and in Fahrenheit degrees button B is pressed.



In order to obtain a value more close to the real one, you should compare the value from micro:bit with a value from a real thermometer. Then the program may be changed by subtracting the difference from the number that micro:bit shows.

Design Challenge 7: Compass

The next application displays which cardinal direction the micro:bit is pointing. After copying the hex file on the micro:bit drive the chip will ask for calibration. For this you will have to tilt the micro:bit in all directions until all the LEDs are on. You will know that the calibration succeeded when a happy face is displayed.



More Challenges to do

1. Build a step counter.
2. Build your own application having in mind the subject you teach!

Resources

<https://makecode.microbit.org/>

<https://makecode.com/labs>

<https://makecode.microbit.org/projects/>

<https://www.itpro.co.uk/desktop-hardware/26289/13-top-bbc-micro-bit-projects>

<https://www.101computing.net/category/bbc-microbit/>

<https://support.microbit.org/support/solutions/articles/19000080171-moving-between-scratch-and-makecode>

References

<https://makecode.microbit.org/projects/>

<https://support.microbit.org/support/solutions/articles/19000080171-moving-between-scratch-and-makecode>

“Whether you want to uncover the secrets of the universe, or you just want to pursue a career in the 21st century, basic computer programming is an essential skill to learn.”

- Stephen Hawking, theoretical physicist and cosmologist