



SciSkills 2.0 Inspirational Booklet.



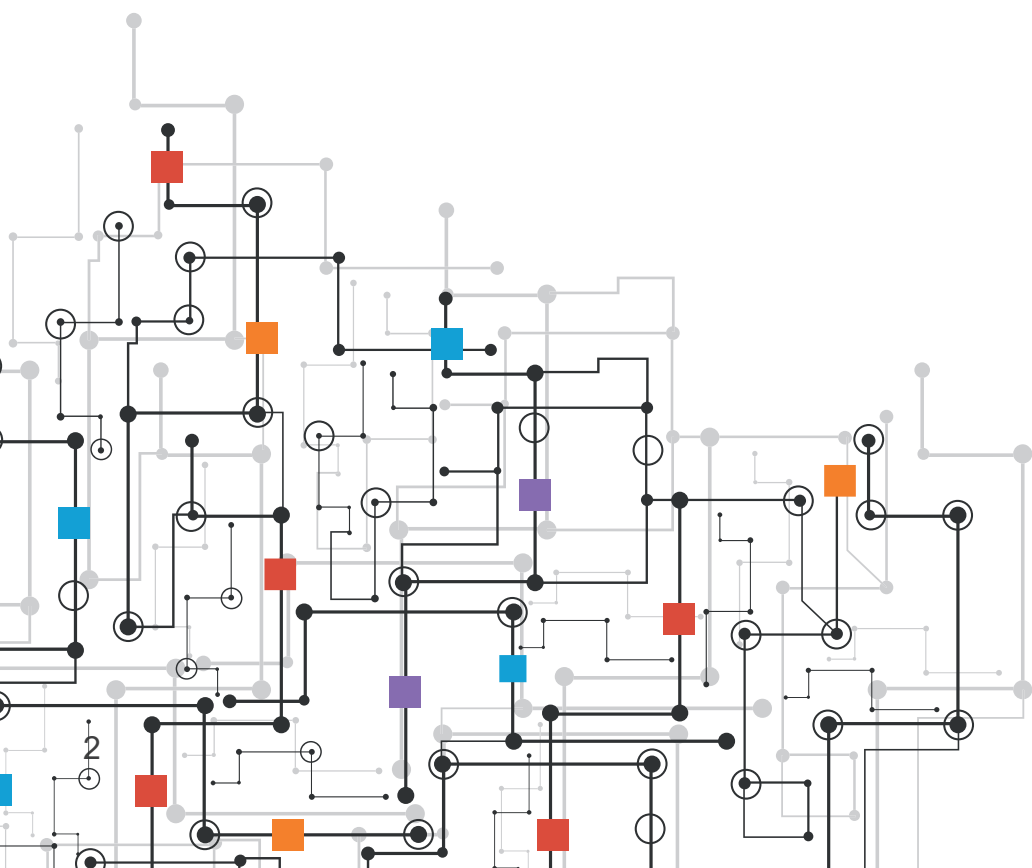
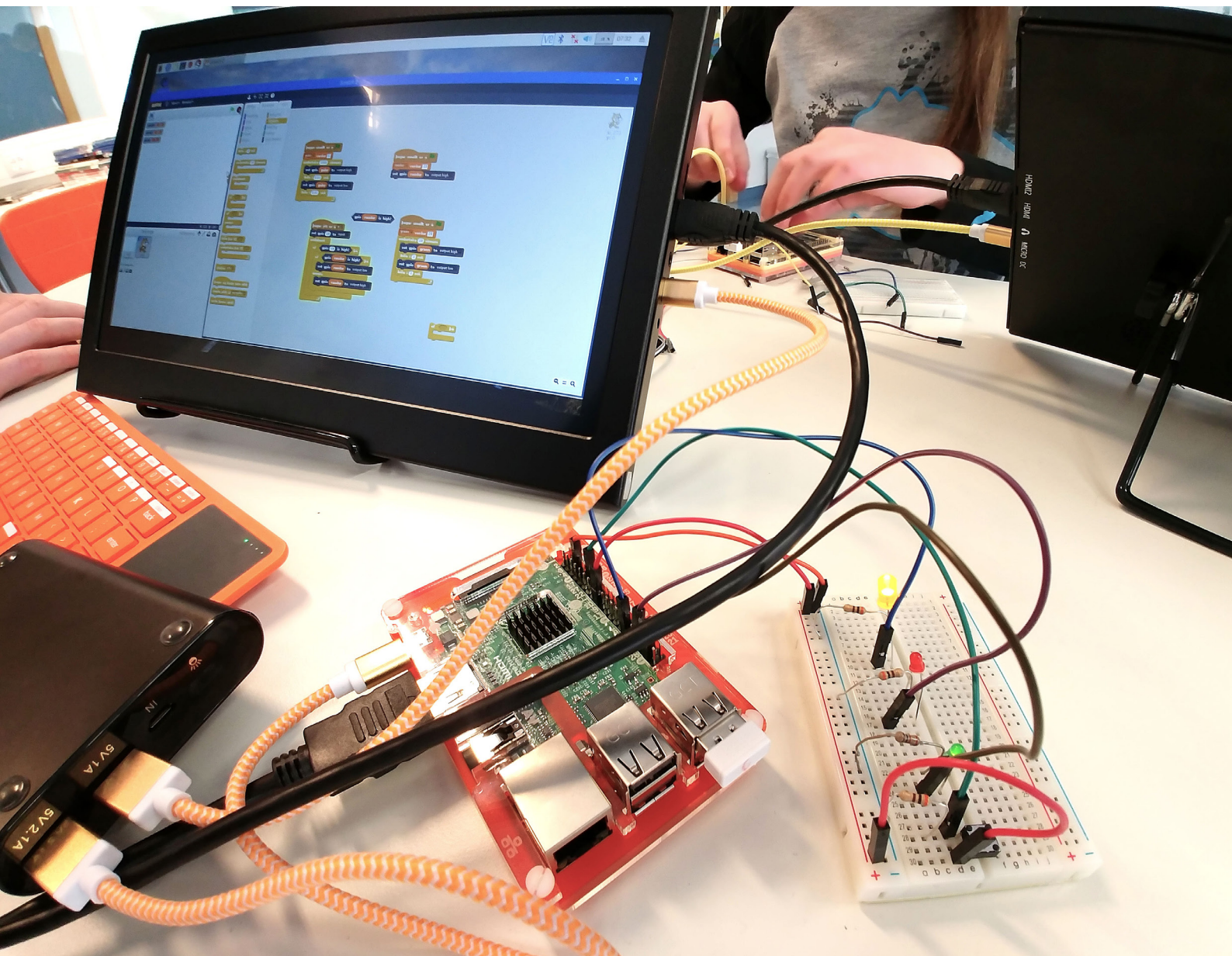


Table of Contents

INTRODUCTION	4
CODING TOOLS & ACTIVITIES	6
Analog programming	6
Blue-Bot	8
Scratch.....	10
Raspberry Pi	14
Micro:bit	17
Digispark	19





INTRODUCTION

Why? For whom?

This inspirational booklet is aimed at teachers and other educational specialists interested in learning about hands-on coding activities offered in Estonia, Sweden and Iceland by Science Centre AHHA, Teknikens Hus and University of Iceland.

The booklet contains the following:

1. Overview of the 6 coding-related activities tested by the three partners during teacher training sessions in 2018, including follow-up tasks and further reading suggestions;
2. Quotes of teachers participating in the teacher training sessions;
3. Short summary of the Nordplus Horizontal project SciSkills 2.0 that made testing and developing the coding activities possible.

Short overview of the activities featured in the booklet

This booklet contains **examples of 6 coding activities** tested during the pilot teacher training sessions organized by the project partners and links to relevant extra resources. All activities are hands-on and most require the use of specific software and/or hardware.

The 6 coding activities are the following:

1. Analog programming – coding without a computer
2. Using Blue-Bot robots
3. Learning a block-based coding language with Scratch software
4. Coding with Raspberry Pi software and hardware
5. Programming with Micro:bit software and hardware
6. Programming with Digispark microprocessors

Quotes from teachers

'We have a "skills training" class at school, during which we also teach how to use the computer and I can use the new knowledge there very well' (teacher from Estonia)

'A terrific course that gave insight into stuff that one can put to use immediately at work. Also it shows the possibility of using coding in all subjects (and it does not need to say "coding" on the timetable. But it was far too short, I'm looking forward to another course! Many thanks!' (teacher from Iceland)



Overview of SciSkills 2.0

The **SciSkills 2.0** project is a trilateral collaboration project between Estonian, Swedish and Icelandic partners lasting from August 2017 till August 2018 and funded by **Nordplus Horizontal**. The project is carried out by Science Centre AHHA from Estonia, Science Centre Teknikens Hus from Sweden and the University of Iceland, all of which are institutions involved in offering further training to teachers and communicators of science on the methods of using informal education settings to support formal education goals.

The aim of the project has been to use and share previous knowledge from teacher trainings and programs and add new ideas to **develop a new teacher-training programme focused on computational thinking, coding and programming**. The project has aimed to provide exchange of knowledge and experiences, work together to develop new teacher training modules and test them in each country as a one-day pilot teacher training sessions between in early 2018, and finally share knowledge and lessons learnt from the pilots. One of the results of the project is this inspirational booklet, which is to accompany further teacher training courses on coding offered by the 3 project partners.



CODING TOOLS & ACTIVITIES

Analog programming

Introduction

A computer program is really just a series of instructions that the computer should perform. There are innumerable computer programming languages in existence, each with their strengths and weaknesses. Computer programming is a process that leads from formulation of a computing problem to executable computer programs.

Programming involves activities such as analysis, developing understanding, generating algorithms, verification of requirements of algorithms including their correctness and resources consumption, and coding of algorithms in a programming language.

This can be a little bit difficult to understand so one way of presenting coding and what is all about is to do it in analog forms. Therefore, we can start by programming without computers to learn what programming is and how programmers work.





Hello, World_

Activity description

This project example focuses on analog programs that are easy to start with. We have three activities that are described in the links. Here you can get some ideas how to start with analog programming and some words connected to programming that could be useful for the students to understand.

The first activity is analog coding in the 100-square. The task is for the students to write their own programming language and try it out with a square-map and some game pieces and also let others try the code.

The second activity is about drawing a house. Here is the task to follow the algorithm, draw, and color their painting. Here the students see how necessary it is for the algorithm to be exact.

The third activity is basic words and concepts

in programming. The goal here is to understand basic words and concepts that are important in programming. So here the participants run and test programs and algorithms with detailed step-by-step instructions consisting of Sequence, Alternative, Repetition and Abstraction that underlie how computer programs are constructed.

Sequence means that the order of the commands you give to the computer is of great importance. Just like baking cake, you have to whisk the eggs before putting the paste in the oven. Otherwise, there will be no cake. In this exercise the students will follow the program with their own bodies. For example "HELLO" means we say HELLO.

Materials

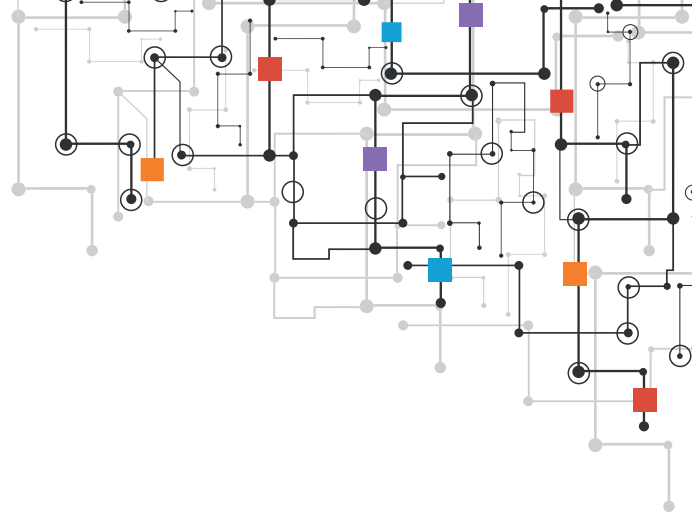
Paper, pens and printed material from the links

Relevant links

Analog programming activities described in detail:

<http://www.teknikenshus.se/partners-projekt/projekt/sci-skills-2-0/inspirationsmaterial/>

Blue-Bot



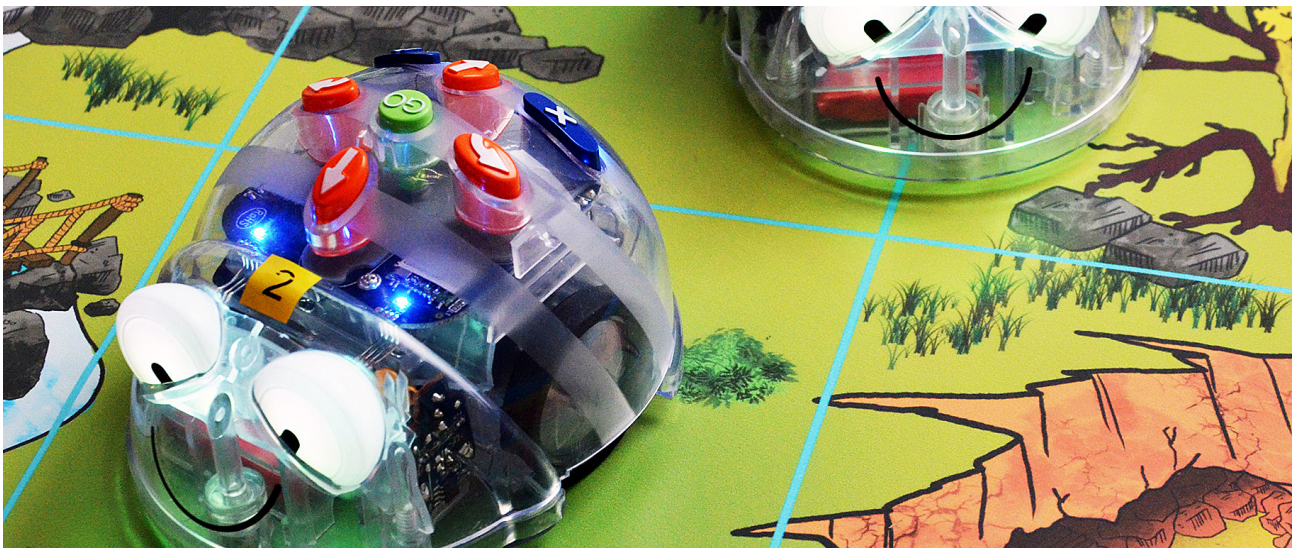
Introduction

Blue-Bots are small robots designed as beetles that are intended to use to test simple programming, mostly in preschool and in the lower ages of the school system, but they can also be used by older children.

They are perfect for exercising logical thinking, digital skills, trouble solving, cooperation and communication. The basics of programming.

The children can work in pairs to learn programming, collaborate and prepare challenges for themselves and for each other.

Blue-Bots are much like Bee-Bots, but equipped with bluetooth, hence the name.



Activity description

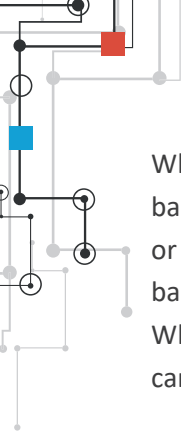
You turn the Blue-Bots on by the switch on the bottom, there you also can turn sound on or off.

Blue-Bots can be programmed directly on the back by pressing the command keys. The buttons are forward, backward, turn left, turn right, pause (1s) and go. The X key clears the memory. Each step the Blue Bot moves is 15 cm. Up to 200 steps can be memorized.

The children can build their own courses or paths,

by drawing or building. Square mats with different backgrounds are also available. The mats have square boxes 15x15 cm. Then the children don't need to measure the distances of the steps, just concentrate on the programming. When programming on the back, no code is visible. One has to remember the code.

The Blue-Bots can also be controlled by using a computer, an iPad or a tablet. The app you use is called Blue Bot and downloaded for free at iTunes, App Store or Google Play. This app can also be used without having a Blue Bot.



When you open the Blue-Bot app you can choose a background, such as the alphabet, geometric figures or a map. You can also use your own pictures as background.

When you connect the Blue-Bot to your iPad, you can no longer control it on the back buttons.

Blue Bots are priced at 80-90€ per piece and 400-500€ per nest (six pieces).

Challenges

- Work in pairs.
- Learn how the Blue-Bot and the app works by trying to connect, explore mode and challenge mode.
- Program your Blue-Bot to move.
- Give your partner a challenge, for example - go from A to B, do not touch the obstacles, make a

pirouette when you reach the goal.

- Work in groups of three or four.
- Everyone has their own Blue-Bot and iPad.
- Program your Blue-Bot to change places with the others, without colliding on the town mat.



Materials

Blue-Bots, iPads, Mats for Blue-Bots

Relevant links

Blue-Bot tutorials

<https://www.bee-bot.us/downloads/file/Getting%20Started%20with%20Blue-Bot%20App.pdf>

<https://hospedagogen.com/lektionsplaneringar-och-ovningar-for-blue-och-bee-bot/>

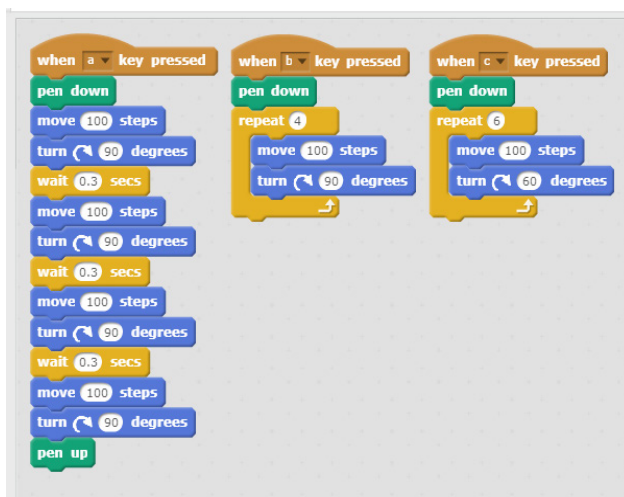
Where to buy

<https://hos.se/produkt/blue-bot-uppladdningsbar/2041>

Scratch

Introduction

There are innumerable computer programming languages in existence, each with their strengths and weaknesses. Some such as Scratch are so-called visual or block-based programming languages in that they are constructed by piecing together blocks of instructions.



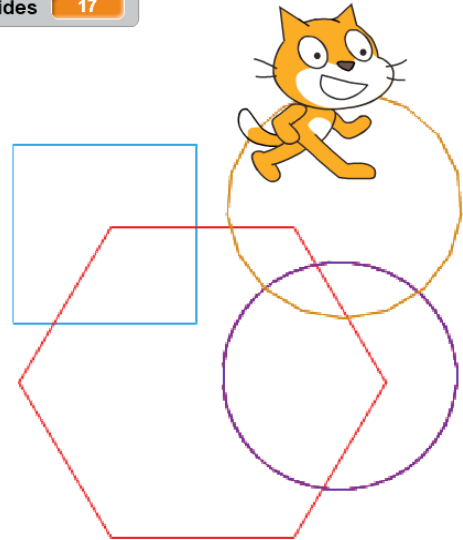
While not replacing text-based programming languages for students wanting to study computer science, block-based programming is a great platform to learn programming. The user has a simple overview of the colour coded command blocks. This clearly shows program structures and enables the user to browse available options and tinker and learn at their own pace on their own terms.

Programming as a tool, rather than a subject

While Scratch is a popular teaching tool for teaching programming through creating simple computer games or animations, it is also a great platform for creating tools for use in various subjects,

The project example below introduces how Scratch can be used in math lessons. Students create their own tools to explore the subject matter in a way that requires them to learn both the subject topic as well as basic programming structures.

number of sides 17



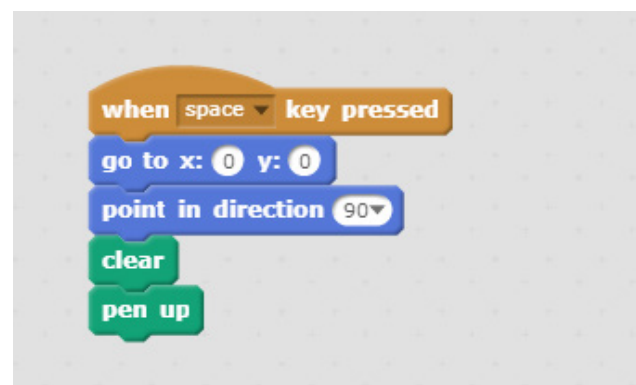
Activity description

This project example focuses on geometry with the student finding which angles the sprite or moving character must turn by to construct polygons of different numbers of sides.

The project consists of five sub-projects, each of which introduce new concepts of math and programming.

Clear the stage

For complete novices, it is useful to start by helping them set up their first procedure by listing the blocks in order. Start by moving the sprite to the centre of the stage, and clearing it.



Draw a square

This first drawing consists of a square with a side length of 100 |

1. Pick an event to start the program
2. Put the pen down
3. Move the cat 100 steps
4. Turn the cat 90°
5. Move the cat 100 steps
6. Turn the cat 90°
7. Move the cat 100 steps
8. Turn the cat 90°
9. Move the cat 100 steps
10. Turn the cat 90°

Once this is done we will have drawn a full square and returned

Draw a square

To have the student get used to finding the relevant blocks, provide them with a pseudo-code (textual description of each block, line by line) as a recipe for which blocks to assemble and in what order.

At this stage it is wise to keep the maths concepts basic; the square has 90° angles which the students are probably already familiar with, helping them connect the code to known concepts.

Draw a hexagon

The student now constructs a program without any pseudo-code, having to figure out that they must adjust the angle by which the sprite turns.

The students must identify the patterns that need to be repeated to add the extra sides and corners and figure out the angle by which the sprite must turn to make the wider internal angle.

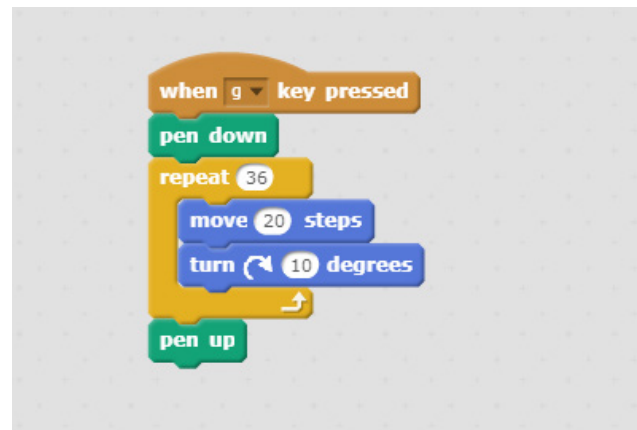
Draw a triacontakaihexagon

Having identified the two blocks (move and turn) that are needed to add an extra side to the polygon, the student is now effectively able to draw a polygon of arbitrary many sides. As the number of sides increases, this starts to become tedious.

Such is the case with the triacontakaihexagon; the 36-sided regular polygon.

Rather than drag more and more pairs to the program, a good programmer will identify the pattern and employ a loop to repeat it once for each side of the polygon.

This vastly simplifies the program.



Draw an n-polygon

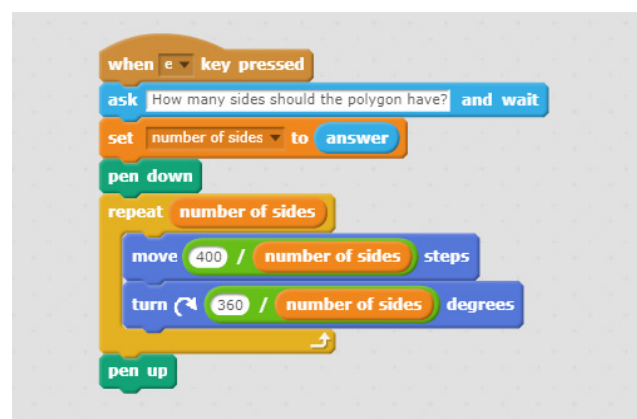
The fifth and final problem is also the one that involves the most advanced topics.

The task is to draw an n-sided (n being able to take on any number: 4-sided, 6-sided, 36-sided...) polygon. To do this the student has to find the general rule to calculate the sprite turning angle from the number of sides/corners.

It is also useful to add a calculation for the side-length. There are various methods for solving this and it can be used as a discussion topic amongst the students.

As the angle appears twice in the code (once in the repeat loop block, and once in the calculation for the turning block) it's useful to store it in a variable.

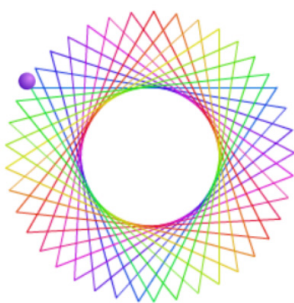
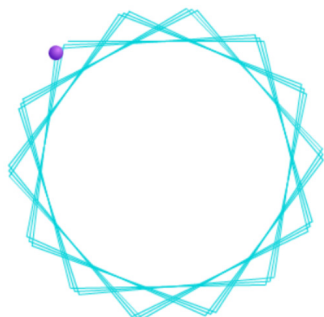
To read in the number of sides, the program can then ask the user for the number of sides. The student now has a fully functioning polygon drawing program that can draw any regular polygon.



Follow-up assignments

Rotation, colours and stars

In the problems above, the students were tasked with drawing precise polygons where they had to figure out the precise angles. If the angles are off by a single degree, the sprite will



miss the starting point, leaving the polygon open.

This, however, can be used to make interesting patterns. Picking an angle at random and cranking up the number of

sides, the program will draw star-like patterns. Playing around with the angles and number of sides, the student can explore various patterns.

The sides can also be coloured differently with the “change pen color by” block.

Additional information

Studios and logging in

Having students create accounts and log in allows them to save their projects on their accounts, share them with project partners and others in the group. Projects can also be organised into studios where they can remix each others’ projects.

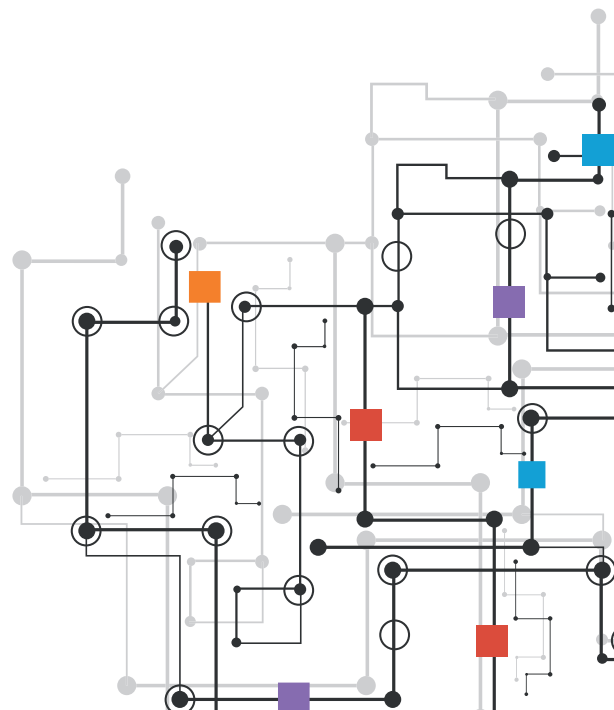
Structure vs. free play

For introductory projects, detailed step-by-step instructions are very useful to introduce the student to the tools. As the student gains in ability it is important, however, to slowly remove the scaffolding. As students gain proficiency, they must be given ample time for free play. It can be daunting to suddenly have to innovate when much of class work

is heavily structured. It can therefore be useful to also provide scaffolding for such free play.

One way of progressing to student-led free play is to have students:

1. choose between a set of tasks (e.g. the geometry task above, constructing a Sierpinsky triangle, draw geometrical representations of multiplication of two numbers, etc...)
2. expand on these projects (e.g. add colours, construct different fractals, draw representations of other mathematical operations)
3. solve problems without guidance (e.g. create a quiz/game to test the user on some math skill, create a drawing program that allows the user to stamp pictures onto an inclined stage and sizes the imprints so that they fit the perspective - smaller ones being further away, etc.)
4. come up with their own problems for them (or each other) to solve



Relevant links

Scratch website and editor: <https://scratch.mit.edu/>

Scratch Wiki with a lot of useful information: <https://en.scratch-wiki.info/>

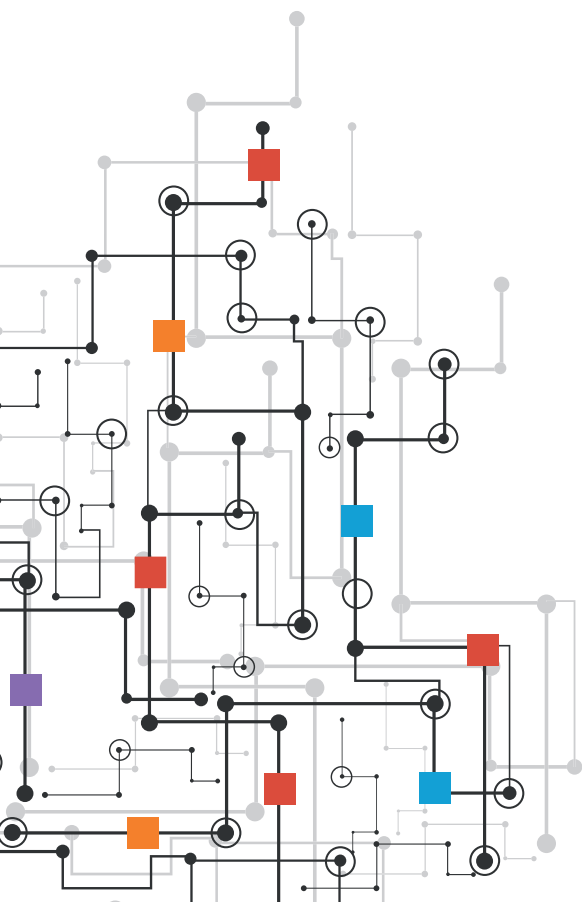
Maths tutorials: https://en.scratch-wiki.info/wiki/Scratch_Wiki:Table_of_Contents/Math_Tutorials

Scratch studio examples: <https://scratch.mit.edu/studios/4258170/>

Polygonal Prowess project: <https://scratch.mit.edu/projects/73456942/>

Quotes from teachers

"We use Scratch in different subjects, to teach programming. We have already used it, the kids are very interested." (teacher from Estonia)



Raspberry Pi

Introduction

The Raspberry Pi is a very useful tool to introduce the student to programming devices in the „real world“ as opposed to just moving characters around the screen. This opens up programming as a problem solving tool to students with more diverse interests and is a great addition to makerspace activities.

The Raspberry Pi is basically just a computer with an operating system written on a MicroSD card. This makes it easy to write a new operating system with all the necessary files onto a card and wipe them clean before each course. The standard Raspbian OS comes with plenty of free and open source tools for an easy start and endless options.

The greatest hurdle which physical computing faces compared to in-software programming, seems to be the extra layer of unfamiliarity. The activities mentioned here are chosen to ease both teacher and student into the world of physical computing through the simple block-

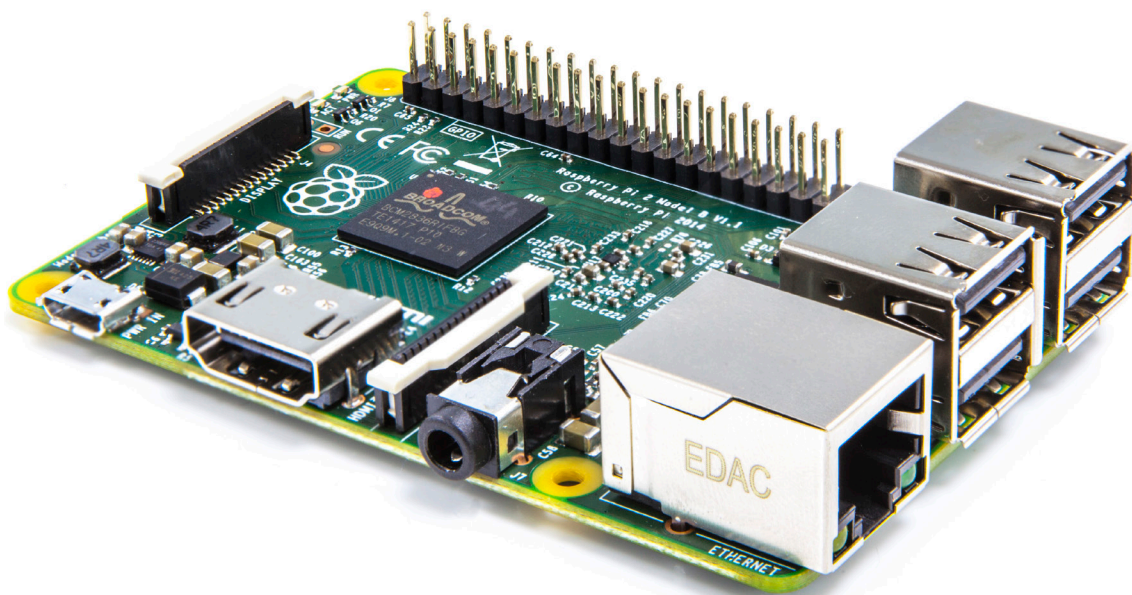
based programming language Scratch which is used to control simple devices such as LEDs through special pins on the Raspberry Pi single-board computer.

Activity description

There are several resources out there with physical computing activities. There are several versions of Scratch and unfortunately each has its own syntax. Each has its strengths and weaknesses, but for the sake of future compatibility and ease of jumping between Scratch physical computing and other Scratch projects, we recommend **Scratch 2**.

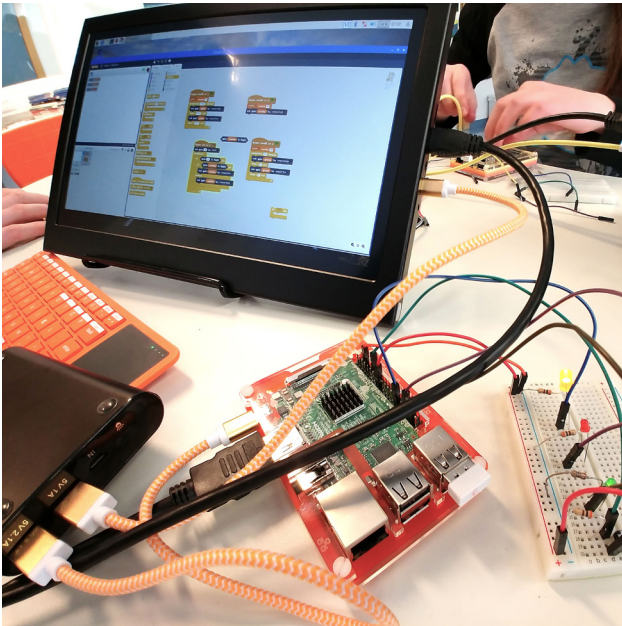
Worth mentioning are also **Scratch 1.4**, especially since it's still better documented and has this useful tutorial (good to look through even if one is sticking with Scratch 2).

Also worth mentioning is the **Scratch GPIO project**, a patched version of Scratch 1.4, but with special tools for interacting with the GPIO pins.



Whichever tool you choose, for this project you will need:

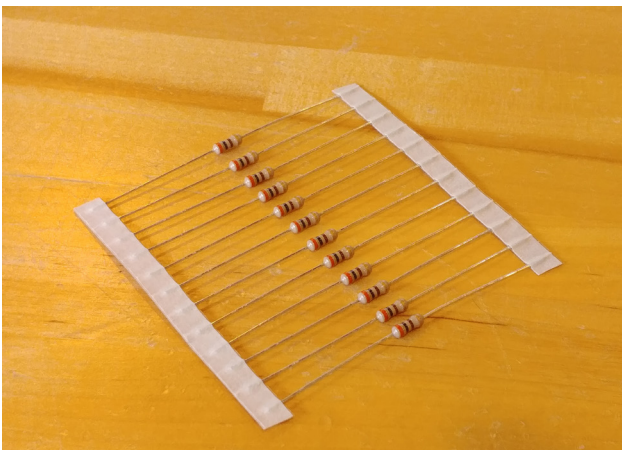
A Raspberry Pi with the Raspbian operating system, a monitor, keyboard and mouse



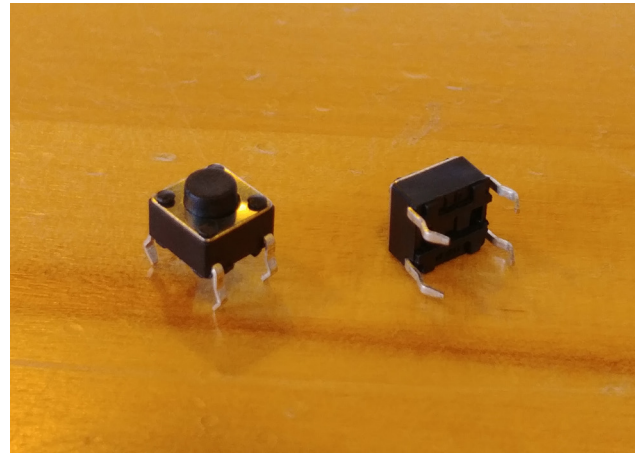
LEDs



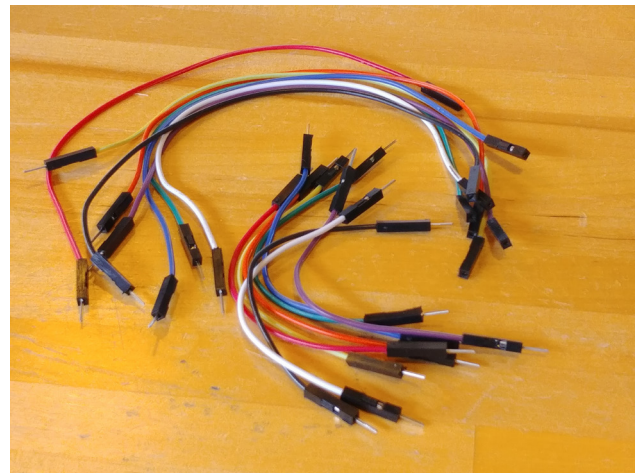
Resistors (ca. 330 Ohm)



A push button



Jumper cables



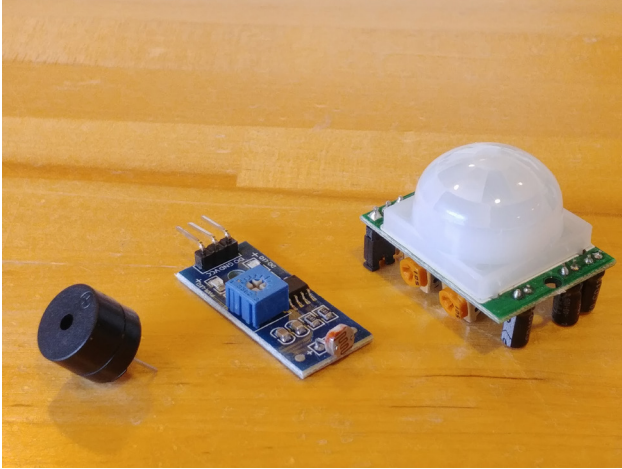
A breadboard



The tutorial ends with connecting a buzzer and a PIR sensor, but the skills acquired by that point are sufficient to start solving own problems and experimenting with these new tools before continuing to new projects.

Follow-up assignments

The button can be replaced with several types of sensors such as the PIR sensor already mentioned or a simple photoresistor.



There are multiple devices that the Raspberry Pi can control instead of the LEDs. They can be directly replaced with buzzers but motors generally need to be controlled through a relay or a controller.

Additional information

When preparing for a course, you may want to have the operating system and available files set up in a certain way. Cleaning out any files that the students may have created and programs that may have been installed or updated can be automated by:

1. Setting up a Raspberry Pi with the operating system and files that you want at the start of a course.
2. Create an image of the Micro SD card (this takes a snapshot of the operating system and all files on the disk) by reading the contents of the card to a separate computer (a desktop or laptop).
3. Write the image file onto one or more Micro SD cards.

Relevant links

The Raspbian operating system download: <https://www.raspberrypi.org/downloads/raspbian/>

Raspbian installation guide: <https://www.raspberrypi.org/documentation/installation/installing-images/README.md>

Controlling DC Motors Using Python With a Raspberry Pi: https://medium.com/@Keithweaver_/controlling-dc-motors-using-python-with-a-raspberry-pi-40-pin-f6fa891dc3d

Scratch 2: <https://www.raspberrypi.org/documentation/usage/gpio/scratch2/README.md>

Scratch 1.4: <https://projects.raspberrypi.org/en/projects/physical-computing-with-scratch>

Scratch GPIO project: <http://simplesl.net/scratchgpio/scratch-raspberrypi-gpio/>
<http://simplesl.net/scratch2gpio/>

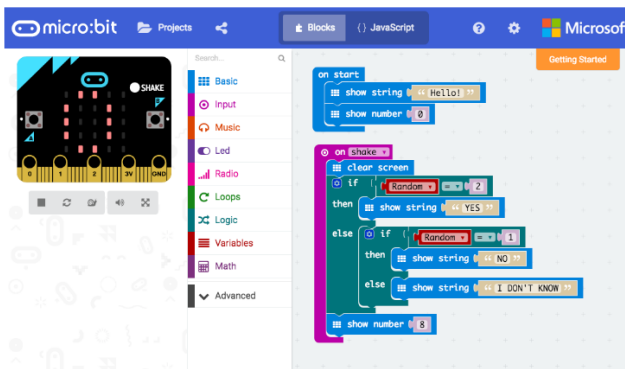
Reading and writing on an SD card on MacOS: <https://medium.com/a-swift-misadventure/backing-up-your-raspberry-pi-sd-card-on-mac-the-simple-way-398a630f899c>

Reading and writing on an SD card on Windows: <https://computers.tutsplus.com/articles/how-to-clone-your-raspberry-pi-sd-cards-with-windows--mac-59294>

Micro:bit

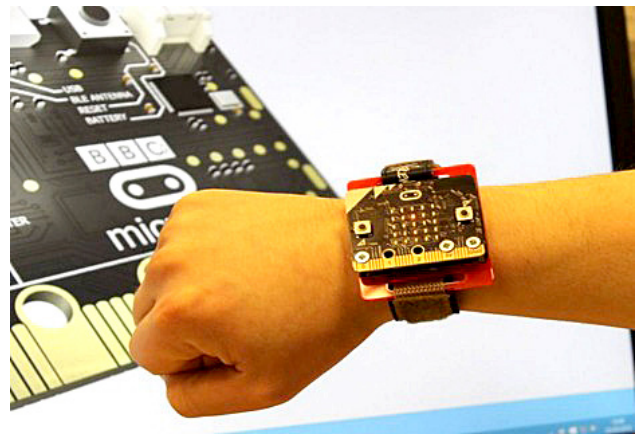
Introduction

Micro:bit is a small computer that has been specifically designed to teach programming to young children. English speaking children can learn rather sophisticated coding already at the age of 11 using the Micro:bit, but others have to wait until they speak better English or have a tutor explain the tutorials to them. But if language is not a problem, then the Micro:bit online ecosystem, that combines block-based programming (similar to Scratch) and step-by-step tutorials, provides one of the easiest options to start learning programming and actually creating sophisticated software and hardware in mere minutes.



Micro:bit can be connected with motors, external sensors, buttons, LEDs like any other development board. Micro:bit also has a LED display capable of showing messages and other information and also buttons to control the device, which make it usable and fun straight out of the box. The device also has a built-in radio and different sensors, such as accelerometer, compass and light sensor, which allow to create very interesting projects, such as Internet of Things devices, wearable electronics, games, remote controlled robots etc.

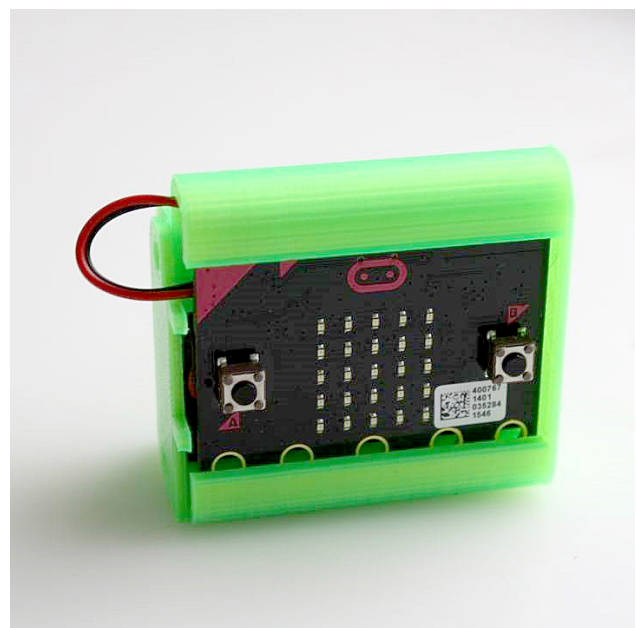
This platform has become so popular that at least 7 countries have given the Micro:bit to their (certain aged) children for free. Similar devices



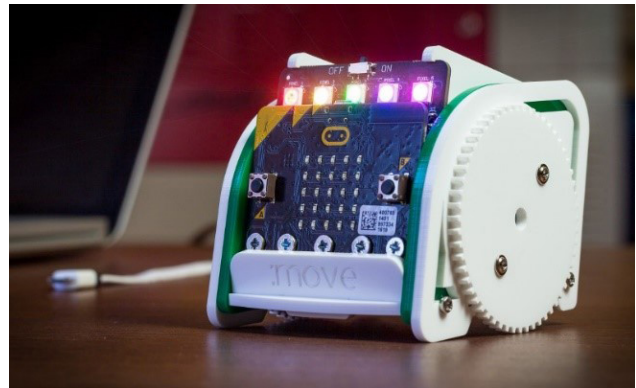
have been developed recently that include the most logical upgrade in their design: a built-in speaker. Their supporting material and tutorials are not as developed, but you might want to take a look at Circuit Playground Express and the German Calliope Mini. The first Chinese imitation, the Newbit, has a multicolor LED display, and the second one, Sino:bit, has a larger screen that can display larger hieroglyphs. There is also a Japanese version called Chibi:bit. Micro:bit is priced between 15-30€, depending on the vendor and the bundle.

Activity description

By now most local electronic stores have it in stock. Programming the Micro:bit is done through their website and you can see a virtual representation of your device. This allows you to test the device even without owning a Micro:bit. You can even program



your Micro:bit via Bluetooth on the go using the Android or iOS app, although it is much easier to program it on a large screen using a mouse.



Materials

Micro:bit device, micro-USB cable, 2xAAA battery pack with JST 2-pin connector, 2xAAA batteries (all of these are included in the Micro:bit Go bundle), a PC, Mac or Linux computer (to program the device), Internet connection (to launch the online programming platform)

Optional

Robotic kits, servo motors, jumper cables, crocodile cables, headphones or buzzers, 3D-printed or handcrafted casing, etc.

Relevant links

Micro:bit homepage: <http://microbit.org/>

Kitronik Accessories for Micro:bit

(robots, sensors, LEDs and more): <https://www.kitronik.co.uk/microbit.html>

Micro:bit Android app: <https://play.google.com/store/apps/details?id=com.samsung.microbit>

Micro:bit iOS app: <https://itunes.apple.com/ee/app/micro-bit/id1092687276?mt=8>

Micro:bit vs Calliope Mini Comparison: https://medium.com/@hello_16463/micro-bit-vs-calliope-mini-160015182c41

Example of Using Radio Communication Code: <https://www.rapidonline.com/bitbot>

Micro:bit 3D-printing Resources: <https://www.myminifactory.com/category/bbc-micro-bit>

Quotes from teachers

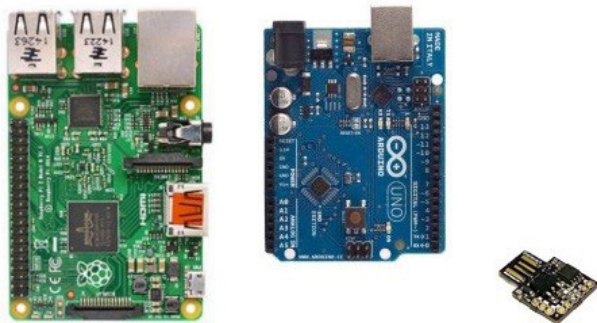
"I will use the Microbit in the future certainly (probably already next school year) in my work. I already got myself my personal Microbit." (teacher from Estonia)

"Programming is a skill that, to different extents, everybody has to know these days. Teachers have to be taught how to teach it to children and they have to be convinced through this kind of training that it is not complicated at all." (teacher from Estonia)

Digispark

Introduction

Digispark is a development board, much like Raspberry Pi or Arduino Uno. However it is even smaller, about the size and weight of a large coin, but also has less capabilities.



Nevertheless, you can call it, in a simplified sense, a small computer. You can use the USB-port of the Digispark to give it power, run computer code that is stored in the device's memory and use metallic pins to connect other computer elements to Digispark. Because it is programmed through the Arduino IDE software, it is very similar to Arduino Uno and other hundreds of prototyping boards out there.







The advantage of Digispark is its small size and low price – about 1€. Its lower capabilities and small size make it perfect to let children program it as a computer virus. This might sound worrying, but in

the modern world it very important to understand how a computer virus and malicious software works and is transferred, which is often via an innocent looking USB-stick. In this case, we won't use Digispark as a small computer at the heart of your electrical prototype (like it was originally intended), we will let Digispark pretend that it is a computer keyboard and send pre-programmed commands to the computer, as if it was a real person.

Activity description

After setting up the Arduino software to recognize Digispark, it is rather easy for children to program their device to imitate a keyboard. This is because people use keyboards very often and they already know what will happen if they press certain buttons. You don't have to teach them what logic to use to run a servo motor or what pin to activate to turn on a LED. Instead, you could order the Digispark to:

1. Press  and  to open the Search function;
2. Type „word” and press  to launch Microsoft Word;
3. Press  again to create a new blank document;
4. Type „Your computer has been infected. Transfer 3 euros to EE43101001004005988028 to free yourself.”

Put the pre-programmed Digispark into the USB-slot of your victim's PC, their computer will trust it automatically and then you can watch how horror will come over their puzzled face. And then you will reveal your prank and you will laugh together about it. The program will stop once you remove the Digispark.

You could make the Digispark do anything you would normally do with a computer, using keyboard commands.

For example:

- Go to a specific website;
- Lock the computer screen every 5 minutes;
- Type weird haunting messages where ever the cursor is;
- Take a photo using the laptop webcam and set it as the desktop background;

Just let your pupils know that they can't program the Digispark to do anything that they wouldn't dare to do on their friends computer in person. For example, it is not allowed to delete all of your friend's desktop files. We are already used to teaching children that

you cannot use scissors to hurt others. Now we have to teach them to use computer programming responsibly.

```
sketch_Jan08a $
#include "DigiKeyboard.h"
#define KEY_UP_ARROW    0x52
#define KEY_DOWN_ARROW  0x51
#define KEY_LEFT_ARROW  0x50
#define KEY_RIGHT_ARROW 0x4F
#define KEY_PRINT_SCR    70
#define KEY_TAB          43
#define KEY_DELETE       76

void setup() {

}

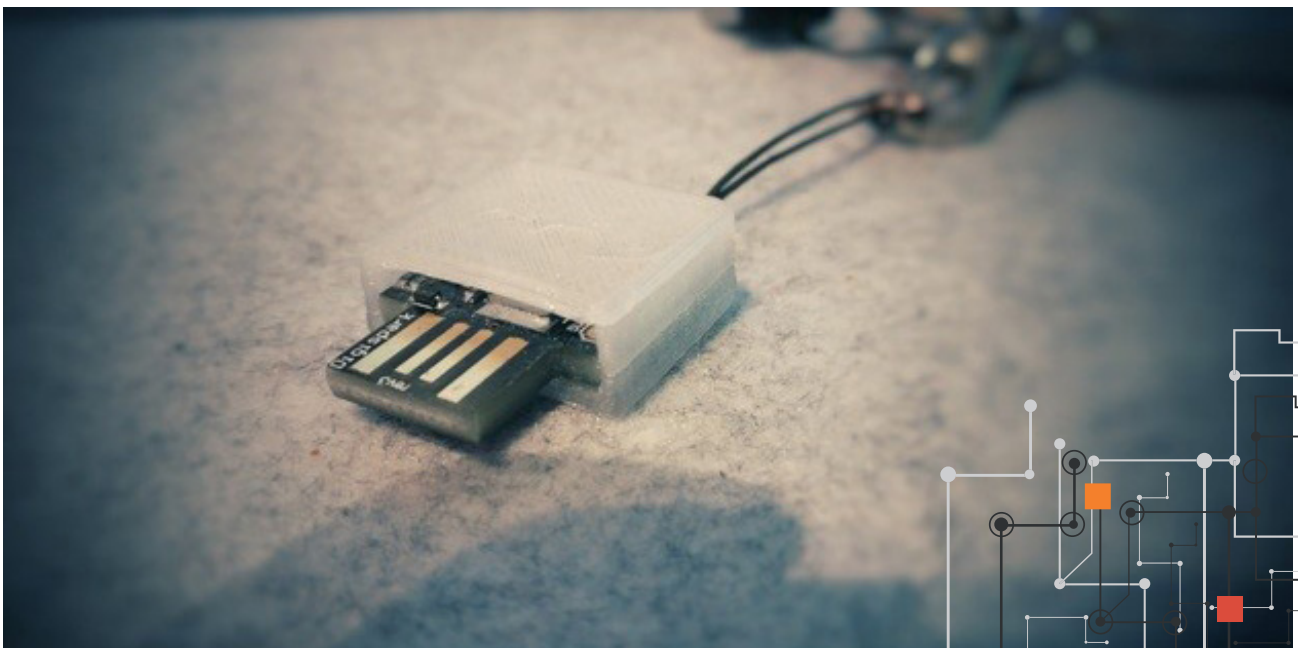
void loop() {
  DigiKeyboard.sendKeyStroke(KEY_R, MOD_GUI_LEFT);
  delay(200);
  DigiKeyboard.println("chrome");
  delay(1000);
  DigiKeyboard.println("ahhaa.ee");
  delay(300000);
}
```

Materials

Digispark device, a PC, Mac or Linux computer with Arduino IDE installed (to program the Digispark)

Optional

hand-made or 3D-printed case for the Digispark (to make it look more like a USB-stick)



Relevant links

Connecting Digispark: <http://digistump.com/wiki/digispark/tutorials/connecting>

Programming Digispark Example:

<https://abhijith.live/build-cheaper-version-of-rubber-ducky-using-digispark-attiny85/>

Buying Digispark from Aliexpress:

[aliexpress.com/wholesale?catId=0&initiative_id=SB_20180108070425&SearchText=digispark](https://www.aliexpress.com/wholesale?catId=0&initiative_id=SB_20180108070425&SearchText=digispark)

Windows and MacOS Keyboard Shortcuts:

https://en.wikipedia.org/wiki/Table_of_keyboard_shortcuts

3D-printable Digispark Case:

<https://www.myminifactory.com/object/digispark-attiny85-badusb-fake-usb-memory-case-40605>

Digispark Homepage: <http://digistump.com/products/1>

Rubber Ducky Malituous USB Explanation: <https://hackmag.com/security/rubber-ducky/>

Contents of the DigiKeyboard.h Library:

github.com/digistump/DigisparkArduinoIntegration/blob/master/libraries/DigisparkKeyboard/DigiKeyboard.h

DigiKeyboard Scancodes: <https://digistump.com/board/index.php?topic=2289.0>

USB Human Interface Device Manual: http://www.usb.org/developers/hidpage/Hut1_12v2.pdf

Digispark Upgrade Project: <http://www.redteamr.com/2016/08/digiducky/>

Rubber Ducky Payload Inspiration for Digispark:

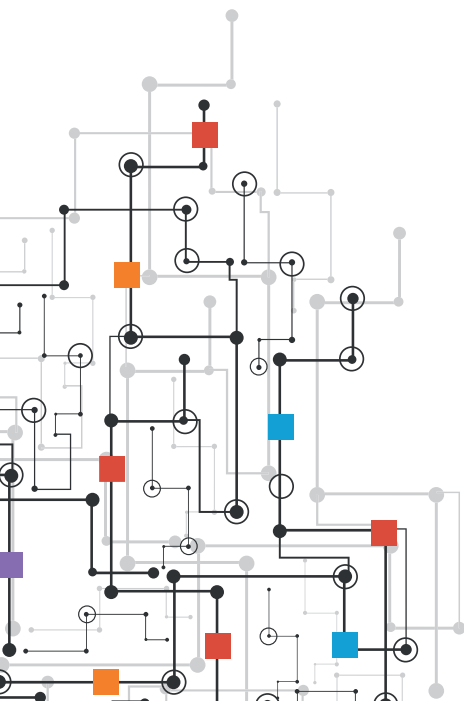
<https://github.com/hak5darren/USB-Rubber-Ducky/wiki/Payloads>

Rubber Ducky Payload to Digispark Converter (beta): <https://nixu-corp.github.io/>

Quotes from teachers

"It is always interesting to do a prank :D" (teacher from Estonia)

*"Digispark can be used to illustrate the importance of cyber defence and how it changes society."
(teacher from Estonia)*









HÁSKÓLI ÍSLANDS



Nordplus



Teknikens Hus

SciSkills 2.0 project has received funding from the **Nordplus Horizontal** program.

Visit our websites for a **printable** or **digital booklet** and other information:

www.ahhaa.ee/meist/ahhaa-projektid/sciskills-2-0

www.teknikenshus.se/partners-projekt/projekt/sci-skills-2-0/

http://visindasmidjan.hi.is/sciskills_20

