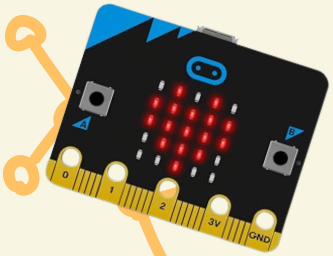
MERCEDES ORBANEJA

Eco-Robotics micro:bit- Eat Fish



Summary:

This **micro:bit** program incorporates a **rock-paper-scissors** game alongside its primary function of **raising awareness about ocean pollution**. The program uses inputs from buttons and gestures to control the game and simulate scenarios depicting the impact of plastic pollution on marine life. Understanding these inputs and functions will help children grasp the consequences of plastic pollution in a fun and interactive way.

Feel free to ask if you need further clarification or assistance with any part of the code! 😋

Our Mission:

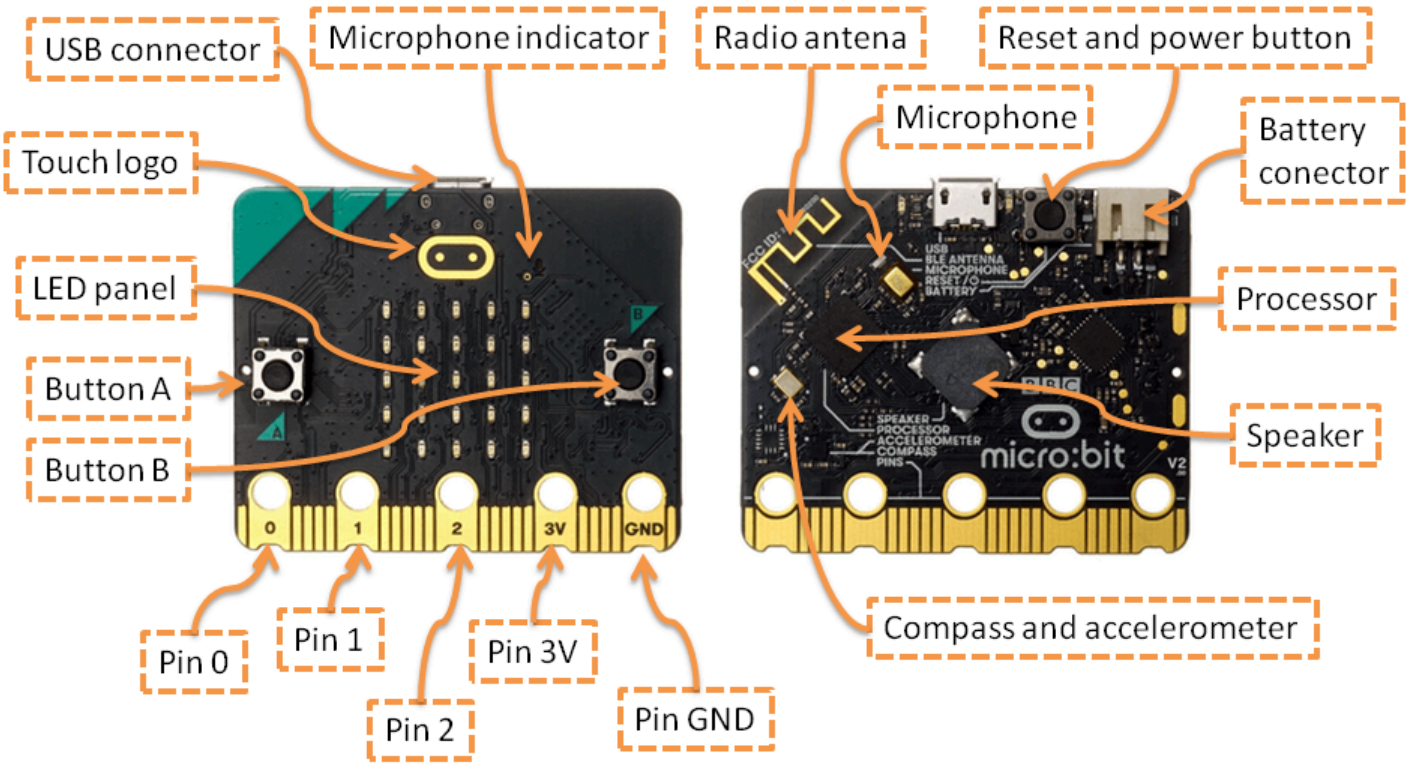
Our mission is twofold: first, to **raise awareness about the importance of protecting the oceans** and how **plastics** harm marine life; second, to educate children about the consequences of plastic pollution through a rock-paper-scissors game. In the game, the fish eats the crab with plastic, the crab cuts the bag, and the bag traps the fish. Both the fish and the crab become contaminated because even though the crab cuts the bag, it still breaks down into microplastics. Through this program, **we aim to empower children with knowledge about the impact of plastic pollution on marine ecosystems and encourage them to take action to protect our oceans**

Understanding the micro:bit:

The **micro:bit** is an educational device designed to introduce students to the world of coding and electronics. It comes with various **built-in features** that allow users to create interactive projects and learn about programming concepts.

Key Features:

- **Compact Design:** The micro:bit features a compact design, making it portable and easy to work with.
- **Programmable**: Users can program the micro:bit using visual programming languages like MakeCode or traditional text-based languages like MicroPython.
- **Sensors**: The micro:bit includes built-in sensors such as an accelerometer, magnetometer, and temperature sensor, allowing it to sense and respond to its surroundings.
- **LED Matrix**: It features a 5x5 LED matrix that can display patterns, numbers, and letters, providing visual feedback to users.
- **Buttons**: The micro:bit has two programmable buttons that can be used as inputs for controlling actions in programs.
- **Wireless Connectivity:** The micro:bit has built-in Bluetooth Low Energy (BLE), enabling wireless communication with other devices such as smartphones and tablets. This allows for remote control and data exchange between devices.



Understanding the code:

This code is designed for a **micro:bit**, a programmable device that can be programmed using **MakeCode** or **MicroPython**.

Functions used:

- **music.play()**:

  This function plays sounds or music on the micro:bit. It takes two arguments: the sound to play and the playback mode

- **basic.showLeds()**:

  This function displays LED patterns on the micro:bit's LED matrix. It takes a text string representing the LED pattern to show.

- **basic.showIcon()**:

  This function shows a predefined icon on the micro:bit's LED screen. It takes an argument specifying the name of the icon to display.

- **basic.clearScreen()**:

  This function turns off all the LEDs on the micro:bit's LED matrix, i.e., it clears the screen.

Setup:

- In the setup function, the program initializes various components and waits until a condition is met before proceeding.

Loop:

- The main logic of the robot's behavior is implemented in the loop function. It continuously checks sensor readings and executes corresponding actions based on the detected conditions.

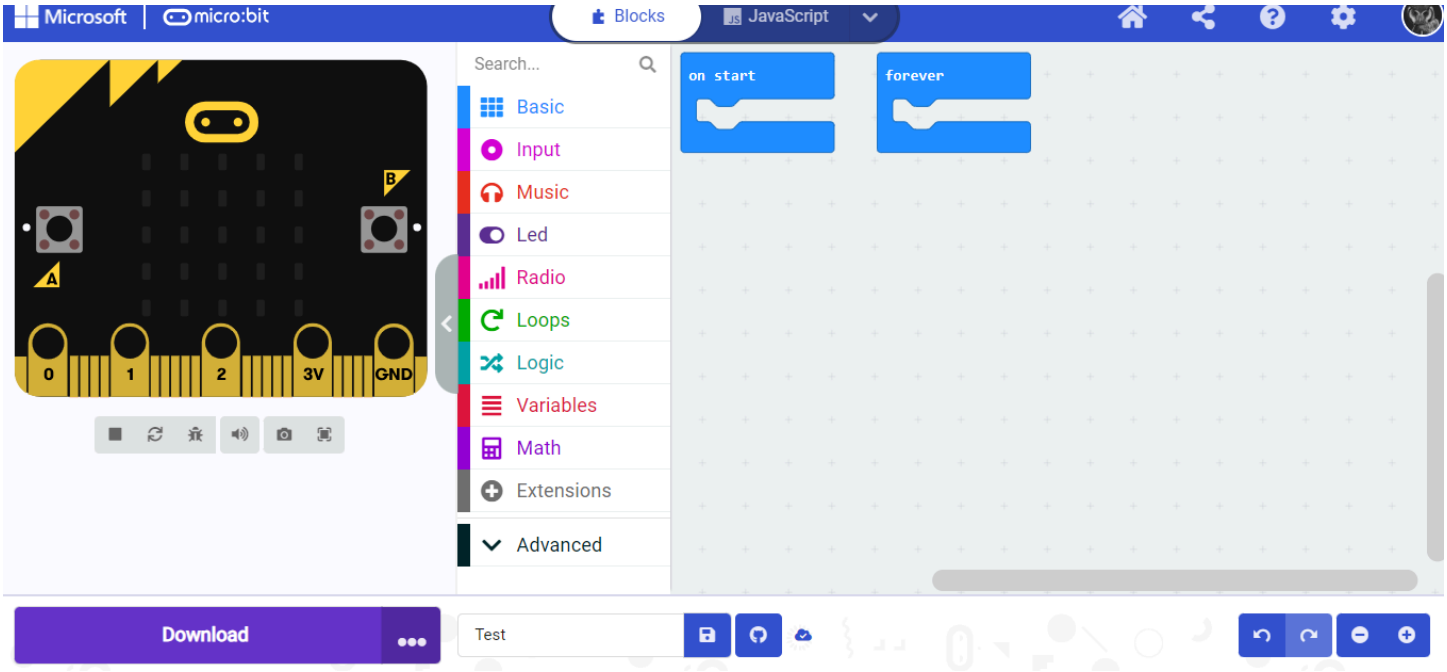
Block programming:

We are going to program through the Microsoft MakeCode online platform:

https://makecode.microbit.org

Interface:

The simulation is displayed at the left, available block at the center column and the programing is at the right. The blocks are dragged from the central column and dropped at the left.



Blocks used for this project:

Basic	1. **on start**: • **Description**: This block defines the code that runs once when the program starts. • **Application**: In the provided code, it's used to initiate playing a sound and displaying an LED pattern at the beginning. 2. **show leds**: • **Description**: This block displays a pattern of LEDs on the micro:bit's LED matrix. • **Application**: In the code, it's used to show different LED patterns representing specific actions or states. 3. **clear screen**: • **Description**: This block turns off all the LEDs on the micro:bit's LED matrix, clearing the screen. • **Application**: It's used to clear the screen before showing new LED patterns. 4. **pause**: • **Description**: This block pauses the execution of the program for a specified amount of time. • **Application**: It's used to introduce a delay in the program execution, allowing time for actions such as displaying LED patterns or playing sounds.
Input	1. **on shake**: • **Description**: This block triggers an event when the micro:bit is shaken. • **Application**: It's used to detect the movement of the micro:bit and trigger a corresponding action.
Music	1. **play**: • **Description**: This block is used to play sounds or music on the micro:bit. • **Application**: In the program, it's used to play different sound effects such as laughter or twinkling.
Led	**Not used in this project**
Loops	**Not used in this project**
Logic	1. **if then**: • **Description**: This block evaluates a condition and executes a set of instructions if the condition is true.

	• **Application**: It's used to make decisions based on different conditions, such as the result of a random number. 2. **else**: • **Description**: This block is used in conjunction with an "if" block to specify a set of instructions that will be executed if the condition is false. • **Application**: In the code, it's used to provide different actions based on whether a condition is met or not.
	1. **set variable**: • **Description**: This block is used to create and set the value of a variable. • **Application**: In the code,the variable “azar” is created to store a number between 1 and 3 selected randomly.
	1. **pick random**: • **Description**: This block generates a random number within a specified range. • **Application**: In the code, it’s used to pick the number between 1 and 3 which will be stored in the “azar” variable.

By **categorizing** the blocks and including the usage of variables, this guide provides a comprehensive overview of the mBlock components and their applications within the provided code. **If you have any questions or need further clarification on any block or concept, feel free to ask!**

Programming:

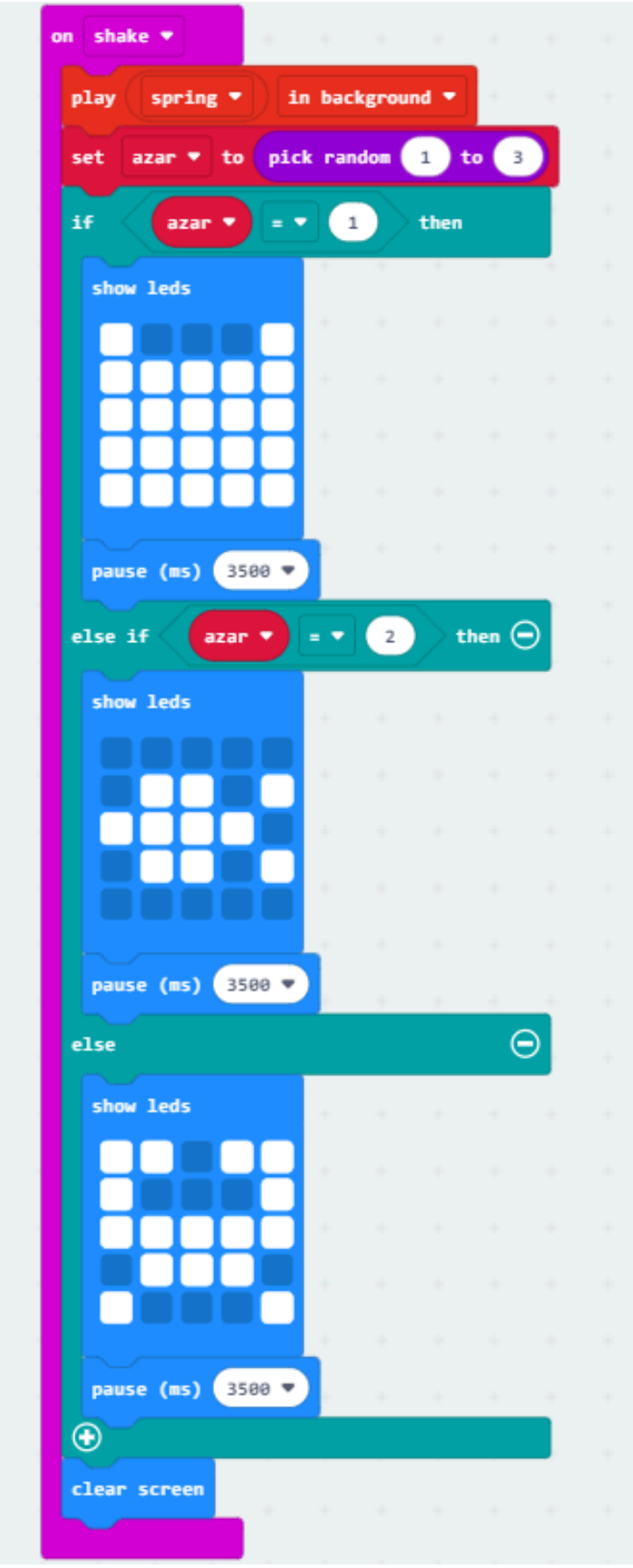
To implement this code using MakeCode visual programming interface:

1. Follow the steps given by the trainner, don’t worry, you have the complete block program below in case you get lost 😅
2. Drag and drop the appropriate blocks onto the workspace.
3. Use control blocks to create conditional statements based on sensor readings.
4. Combine movement blocks with these conditions to control the robot's behavior 🤩
5. Utilize the delay block to introduce pauses or timing constraints as required.

By assembling these blocks and configuring them according to the logic outlined in the code, you can replicate the functionality in MakeCode visual programming environment.

Feel free to reach out if you need assistance with specific blocks or their configurations!

Block program for game tipe rock-scissors-paper:



That's all:

I hope you enjoyed the workshop 💚

Mercedes Orbaneja Fernández
Secondary Education Science Teacher
Electric Engineer

Further informacion and tutorials:

The micro:bit, developed by the BBC, is an educational tool designed to teach coding and digital skills to students. It supports educators by providing an engaging and accessible platform for teaching programming, integrating into various subjects, offering professional development resources, and fostering project-based learning. Please visit the official site for further information, tutorials, class engagement and community.

https://microbit.org