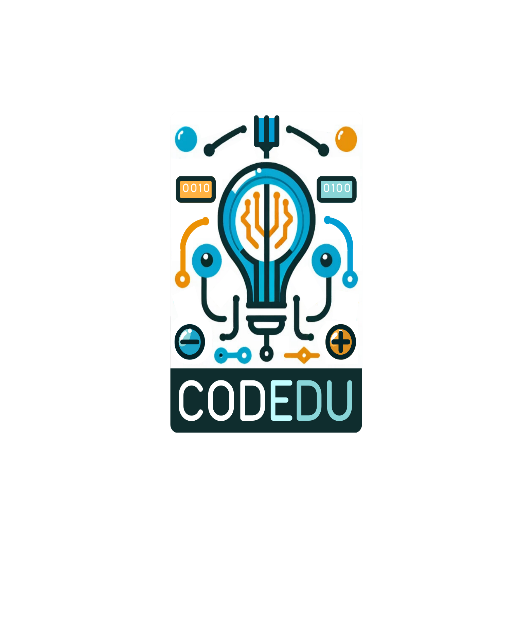
[[1]](#footnote-1)



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CODEDU’s Teachers’ Training Curriculum** | | | | |
| **Section 4: Practical Introduction to Arduino** | | | | |
| **Subject:** Arduino | **Duration (in hours):** 5/6 hours | |  | |
| **Target audience:** teachers | | | | |
| **Training methodology:**   * Project-Based Learning (PBL) * Online simulations using Tinkercad * Hands-on experimentation with Arduino kits * Collaborative group work * Practical activities | | | | |
| **Level (and cycle, if applicable) of the learning experience:** Introductory course for teachers with no prior experience with Arduino | | | | |
| **Assessment method:**  Project-based assessment | | **Form of participation in the learning activity:**  In-person and/or online | | |
| **Expected Learning Outcomes:**  • Understand the basic principles of Arduino and its applications in education.  • Develop and program simple circuits using sensors, LEDs, buttons, and motors. | | * Design creative projects that combine electronics and coding to solve real-world problems. * Utilize online simulation tools (such as Tinkercad) to prototype and test circuits. * Gain confidence in teaching Arduino-based activities and guiding students through project development. | | |
| **Prerequisites needed to enrol in the learning activities (if needed):** | | | | |
| **Supervision and identity verification during the assessment:** | | | | |
| • Unsupervised with no identity verification. | | | |  |
| • Supervised with no identity verification. | | | | **X** |
| • Supervised online or onsite with identity verification. | | | |  |

|  |
| --- |
| **Module 4.1** |
| **Arduino Projects for Beginners** |
| Electronics Concepts  What are Electronics?  Electronics is a field of science focused on designing and developing circuits that utilise electrical and electronic components to represent, store, process, or transmit visual or mechanical information.  Voltage  Voltage, also referred to as potential difference (PD), is the measure of the electrical potential difference between two points or the difference in electric charge. The unit used to measure voltage is the volt. To measure voltage, a voltmeter is used, with the ground (neutral) serving as the reference point.  Electric Current Electric current is the steady flow of charged particles, which refers to the movement of electric charge through a conductor (such as copper) when there is a potential difference between the two ends.  Direct Current Direct Current (DC) is a type of current where electrons flow consistently in one direction. It is typically generated by small power sources such as batteries (9V), rechargeable batteries (1.2V and 1.5V), dynamos, solar panels, and devices that convert alternating current (AC) to DC. DC is commonly used to power electronic devices (ranging from 1.2V to 24V) and has a positive and a negative terminal.  Alternating Current Alternating Current (AC) is an electrical current that periodically reverses its direction, unlike direct current (DC), which maintains a constant flow in one direction.  Theoretical concepts  Electronics and robotics are continuously advancing, and programming evolves at the same rate to keep up with emerging technologies. Therefore, our journey in programming should start with understanding foundational concepts such as algorithms, logic, and others. These concepts are essential across all areas of programming and form the basis for effective learning in the field.  Algorithm  An algorithm is a finite sequence of properly defined and unambiguous instructions, each of which is mechanically performed in each finite period of time. The algorithm concept is often illustrated with the example of a recipe, although there are much more complex algorithms. They can repeat steps (iterate) or require decisions (such as logic or comparison) until the task is finished. An algorithm is not necessarily a computer program, but the steps needed to accomplish a task. Its implementation can be done by a computer, by some sort of robot or even a human being.  Example:  Flowchart Examples  **Figure 1- Example Algorithm**  Language  A programming language is a language designed to perform operations on machines or computers. It is a structure of rules that allows actions to be performed. The process of writing a computer program is called schedule or sketches.  Arduino  Arduino is an open-source platform used for building electronics projects. It is is a programmable physical circuit board designed with a microcontroller. This device allows it to be programmed through a software or IDE (Integrated Development Environment) that runs on your computer, used to write and load the computer code to the physical board. The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board – you can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program  Sketches  The process of writing a computer program using Arduino is called Sketches. These are written in the text editor of Arduino IDE and are saved with the extension .ino file.  Variables  They may be defined as everything that may affect the actions to be performed. They are used to store read values, which are information that is treated by the microcontroller.  There are several types of variables. For the purpose of this module, three types will be presented.  **- Boolean Variables**  Boolean variables can only have two values: true (true) and false (false).  Boolean running = false;  **- Int**  Int variables are used for numeric storage.  Int ledPin = 13;  **- Char**  Char variables are used for characters storage and must be written between single inverted commas.  Char MyChar = 'A';  Comparison operators  These operators compare the variables to values. The word **if** is used to express a condition and conditions are necessary for the execution of an operation.  When **if** is introduced, the user wants to check is the condition is satisfied or not. If the condition is satisfied, an action will happen. If the condition is not satisfied, another action will take place.  Example:  if(Some Variable> 50)  {  // Code  }  The example above shows that if some variable is higher than 50, then the action that will be performed is the one described in between the cotters. By contrast, if the variable was lower than 50, that action will not happen and another action would be defined.  **Some verification examples with the use of if:**  x == y (x is equal to y)  x! = y (x is not equal to y)  x <y (x is less than y)  x> y (x is greater than y)  x <= y (x is less than or equal to y)  x> = y (x is greater than or equal to y)  Arithmetic operators  These are symbols that represent arithmetic operations. They are applied in the variables.   * (equality)   + (addition)  - (subtraction)  \* (multiplication)  / (division)  % (rest of division)  Structures  Any program is organized in a specific structure, so that is makes sense and runs properly.  Arduino has a predefined structure composed of two main functions: setup and loop.  The ***setup* ( )** **function** is activated (invoked) when the program starts. This function will start your variables, the configuration of the inputs and outputs associated with each Arduino pin, etc.  The representation of this function in the programming language will appear as follows:  setup() {  }  The **loop ( ) function** is activated (invoked) after the setup ( ) function declaring the initial values. The loop ( ) function does exactly what its name suggests, comes into looping (always runs the same code block), allowing the program to constantly respond to the defined programming.  The representation of this function in the programming language will appear as follows:  loop() {  }    Digital functions  These functions are used to configure the inputs or outputs.  **- pinMode ()**  It allows the pin configuration as input (input) or output (output).  Example:  pinMode(9, OUTPUT); // This means that the digital pin 9 was configured as an output.  **- digitalRead ()**  Reads the value of a specified digital pin, which can be HIGH or LOW. Yet the information of High or Low is not visible at this stage of programming.  Example:  buttonState =digitalRead(9); // This means that the microcontroller will read the State of pin 9.  **- digitalWrite ()**  Writing in the programming (sketch) the words HIGH or LOW that are considered as value to a digital pin.  Example:  digitalWrite(9 HIGH); // This means that pin 9 is HIGH.  Analog functions  These functions are used for reading or writing analog values.  **- analogRead ()**  This function allows to read the value of a specified analog pin.  **- analogWrite ()**  This function allows to give an analog value to a specific pin. It can be used to provide the connection instruction varying the LED brightness or allowing rotation of a variable speed motor.  Example:  analogWrite(9,134); // This means that the analog value of 134 was given to pin 9  Components used on Arduino Projects  The Inputs  The inputs are electronic or mechanical sensors receiving and transmitting information based on the read signals (in the form of temperature, pressure, humidity, light, motion, pH, etc.) Examples of inputs are: gas sensors, temperature, LDR (Light Dependent Resistor), motion sensors, among many others.    **Figure 2- Example of an DHT11**  The outputs  The outputs convert current signals or voltage into physically useful signals such as: movement, light, sound, rotation, etc. Examples of outputs are motors, LEDs, buzzer and others.    **Figure 3 - Example of a Buzzer**  Digital signals  Digital signals are characterized by having two different states 0 or 1 (corresponding the 0 to false (F) and the 1 to True (T)). The figure below illustrates a signal in digital format.    **Figure 4 - Digital Signal Example**  Analog signals  These signs are enclosed within a range of values. The limits are defined and the values must be inside the limits. The figure below illustrates a signal in analog format.    **Figure 5 - Example analog signal**  Batteries  Batteries are used to supply power to the circuits.    Figure 6 – Li-ion Battery  One of the most important rules when handling a battery is that we should never connect to the positive (+) and the negative (-) together. If we do, we will cause a short-circuit.  Resistor  The electrical resistor can be defined as a barrier to the passage of electric current in a circuit. The electrical resistance, symbolized by the letter R, is measured in ohms (Ω) and is represented as follows: *R = 100 Ω* ( *100 - is its numerical value* and *Ω - it is the unit in which the measured value is (Ohm))*  The electrical resistor is one of the most versatile electronic components. Its main aim is to limit the passage of electric current in a given circuit. The resistance is a linear component, meaning that when subject to a potential difference, its "answer" is a current that varies linearly according to the applied voltage. The two most common symbols to represent a resistance are as follows:    **Figure 7 - Example resistors symbology**  The resistor value can be determined by the different colors of bands on the resistance’s body, as we can see in the figure below. The first circle gives us the first digit of the resistance value, and the second gives us the second digit, the third circle indicates the number of zeros that must be added to the first two circles to get the correct value. The correspondence between the colors used and the figures are given in the following table:    **Figure 8- Colour coding of resistors**    Breadboard, White Board or Protoboard  A screenshot of a computer  Description automatically generated  **Figure 9 - Breadboard**  Breadboard, White Board or Protoboard is one of the most useful equipment in electronics’ learning, because it is where all connections are made and tested. It consists of a didactic plate comprising a contact array that allows the construction of experimental circuits without the need to perform welding of components. It enables a set of experiments with the same components by inserting or removing them quickly and safely.  A screenshot of a computer  Description automatically generated  **Figure 10 -Example the negative connection plate.**  A screenshot of a computer  Description automatically generated  **Figure 11 -Example positive connection plate**  Power connections must be made to a source of energy that will be used in the assembly. The power buses are two lines indicated in blue (-) and red (+) entry points. Each line has its points electrically connected through a conductive metal support which is below the plastic cover of the breadboard. These buses will provide the energy needed to assembly on the workspace.  A screenshot of a computer  Description automatically generated  **Figure 12 - Workspace**  The workspace consists of two distinct columns with several lines of insertion points where the points of a line are electrically connected to each other, but the lines are isolated from each other, as can be seen in the section shown above.  LED (Light Emitting Diode)  **Figure 13 -Light-Emitting-Diode**  Light-emitting-diode or simply LEDs, are critical components in the world of electronics. Its main feature is the emission of light in electronic equipment, whether as microelectronics products as warnings flag, or in some larger equipment such as traffic lights. Simplifying the concept of LEDs, one can say that they are small lamps with varying colors easily integrated in electrical circuits. So that the LED emits light, it is necessary that the anode (+) is positive with respect to cathode (-), when that happens, the LED is forward-biased, as shown below.  A screenshot of a computer  Description automatically generated  **Figure 14 - Direct Polarization LED**  Potentiometer  The potentiometer allows you to adjust the intensity of current that will pass through the LED, thus allowing to observe a light intensity variation. The potentiometer is an electronic component that functions as a variable resistance. It can be used to measure position, directions, currents, voltages, etc.  The potentiometer can have two schematic representations:  C:\Users\André\Pictures\transferir.jpg  **Figure 15 - Symbology Potentiometer**  And it may look like the images below:  **Figure 16 - Examples of potentiometers**  Button  The button is one of the most used elements in electronics as it is used to enable some action. The buttons are generally activated when pressed. A button on an electronic circuit normally operates as an electrical switch having two contacts therein which when pressed, causes the energy to circulate in the circuit.  A black and white drawing of a bench  Description automatically generatedA group of small black and silver electronic components  Description automatically generated  Figure 17 - Symbology Button  Figure 18 -Examples buttons |

|  |  |
| --- | --- |
| **Activity** | |
| **Activity** |  |
| Practical Concepts  You will need to access the web page [www.tinkercad.com](http://www.tinkercad.com). Where you can perform all the necessary assembly of the circuits described. There is no need to purchase physical materials. Let’s do this step-by-step  **STEP 1**  Access the page [*www.tinkercad.com*](http://www.tinkercad.com)    **Figure 19 - Page tinkercad**    **STEP 2**  Select Sign up for free and make the registration.    **Figure 20 - Registration in tinkercad**  **STEP 3**  Select Circuits - Create New Circuit  A screenshot of a computer program  Description automatically generated  **Figure 21 - Create New Circuit**  **STEP 4**  This is the work area where assembly will be made.  A screenshot of a computer  Description automatically generated  **Figure 22 -Place assembly into the circuit**  **STEP 5**  To do the assembly of the electronic circuit go to the "Components" icon, and insert the components starting with the breadboard:  A screenshot of a computer  Description automatically generated  **Figure 23 -Form inserting components in tinkercad**  **Note: All exercises developed should follow the five previous steps.** |
| **Sources:** |
| **Extra contents:** |

|  |  |
| --- | --- |
| **Activity** | |
| **Activity** |  |
| LDR & LED An LDR (Light Dependent Resistor) is a sensor that changes its resistance based on the amount of light it detects, making it ideal for projects like automatic lights or light-sensing devices.  Learning outcomes   * Condition if / else * Read an Analog Input * Write a digital output   **Materials needed / Assembly**   * Arduino * LED * LDR * Resistor 1 Kohm * Resitor 100 Ohm * Jumpers * White Board   **STEP 1**  Copy this simple schematic into your TinkerCAD Circuit  A circuit board with wires and wires  Description automatically generated  1kΩ  100 Ω  **STEP 2**  Click on the "Code Editor" and copy to the program area the code below. Then click on the "Upload & Run" tab as shown below  A screenshot of a computer  Description automatically generated  **Below is an explanation of the lines of code you have just copied. This knowledge will allow you to understand / recognize future programming lines.**  *int ledPin = 8; // Define the digital pin 8 as the pin where the LED is connected*  *int lightIntensity; // Variable to store the light intensity value read from the LDR*  *int lowIntensity = 150; // Threshold value for low light intensity*  *void setup()*  *{*  *pinMode(A0, INPUT); // Set analog pin A0 as input for reading the LDR*  *pinMode(ledPin, OUTPUT); // Set digital pin 8 as output for controlling the LED*  *Serial.begin(9600); // Initialize serial communication at a baud rate of 9600 for debugging*  *}*  *void loop()*  *{*  *lightIntensity = analogRead(A0); // Read the analog value from the LDR (light intensity sensor)*  *Serial.println(lightIntensity); // Print the current light intensity value to the Serial Monitor*    *if (lightIntensity <= lowIntensity) { // Check if the light intensity is below or equal to the threshold*  *digitalWrite(ledPin, HIGH); // Turn the LED on if the light intensity is low*  *}*  *else {*  *digitalWrite(ledPin, LOW); // Turn the LED off if the light intensity is above the threshold*  *}*  *delay(100); // Wait for 100 milliseconds before repeating the loop*  *}* How it works Press the “Start Simulation” Button in the upper right corner, after that you should be able to interact with the circuit, by pressing the code button, like before you should see your code, now press on the bottom button that says, “Serial Monitor” and it should appear something like this:  A screenshot of a computer  Description automatically generated  Now select the LDR and it should appear a slidebar to set the simulated brightness, and when you set it up to the minimum, simulating the night the LED lights up: |
| **Sources**: |
| **Extra contents:** |

|  |
| --- |
| **Activity (Assessment of Module 4.1)** |
| **Type**: Project-based assessment |
| Based on the knowledge acquired in the training, participants should develop a small functional project using Arduino and sensors to their liking, which can first be simulated in Tinkercad before physical implementation.  Assessment criteria:  Originality of the idea (10%)  Degree of difficulty (20%)  Correct functioning of the circuit (30%)  Well-structured and functional code (30%)  Presentation and explanation of the project (10%) |

|  |
| --- |
| **Module 4.2** |
| **Advanced Arduino Projects** |
| Advanced Electronics Concepts Understanding advanced electronics is vital for designing complex and efficient Arduino-based projects. This section explores key concepts in-depth: Communication Protocols Communication between devices is an essential part of advanced electronics. SPI, I2C, and UART provide robust mechanisms for sharing data between Arduino and peripherals. SPI is particularly effective for high-speed requirements, while I2C simplifies setups involving multiple devices. UART remains the go-to option for basic serial communication, making these protocols foundational tools for any advanced project. UART (Universal Asynchronous Receiver-Transmitter) UART is a widely used communication protocol for serial communication. Unlike SPI or I2C, UART does not require a clock signal, which simplifies the wiring. Instead, it uses two primary lines:   * **TX (Transmit):** Sends data to the receiving device. * **RX (Receive):** Receives data from the transmitting device.   In UART communication, both devices must agree on parameters such as baud rate (data transmission speed), parity, and stop bits. Common baud rates include 9600, 115200, and 57600. Example: Arduino Communication with HC-05 Bluetooth Module Arduino e o Módulo Bluetooth HC-05 • AranaCorpThe HC-05 Bluetooth module is an excellent example of UART communication in action. This module allows wireless communication between Arduino and other devices like smartphones or PCs Advanced Sensors and Modules Sensors and modules expand the capabilities of Arduino, allowing it to interact intelligently with its environment. PIR (Passive Infrared) Sensors A Passive Infrared (PIR) sensor is an electronic device that detects motion by sensing changes in infrared radiation within its field of view. It operates by measuring the heat emitted by objects, such as humans or animals, and detecting movement when the infrared levels change. PIR sensors are widely used in applications like motion-activated lights, security systems, and automated appliances due to their low cost, energy efficiency, and reliability.  In Summary the PIR Sensor does:   * Detect motion by sensing changes in infrared radiation. Ideal for security and automation projects. * Passive Infrared Sensor - an overview | ScienceDirect TopicsApplications include motion-activated lights or alarms.  Ultrasonic Sensor The HC-SR04 is an ultrasonic sensor widely used for precise distance measurement in various applications. It operates by emitting ultrasonic sound waves through a transmitter and then measuring the time it takes for the reflected waves to return to the receiver. This time delay is used to calculate the distance to the object with high accuracy. The sensor has a range of approximately 2 cm to 400 cm, with a resolution of about 3 mm. Known for its simplicity, reliability, and low cost, the HC-SR04 is a popular choice for robotics, obstacle detection, level monitoring, and automation systems.  In summary the Ultrasonic sensor:   * Measure distance by emitting ultrasonic waves and calculating the time it takes for the echo to return. * Widely used in obstacle detection, liquid level monitoring, and distance measurement.   HC-SR04 Ultrasonic Distance Sensor - SparkFun | Mouser HC-05 Bluetooth Module The HC-05 is a Bluetooth module designed for wireless communication between devices. It operates on the Bluetooth 2.0 protocol and supports both master and slave configurations, making it versatile for various applications. The module is easy to integrate and widely used in projects like wireless data transmission, remote control systems, and IoT devices. With a range of about 10 meters, it allows seamless communication between microcontrollers, computers, or smartphones. Its simplicity, low power consumption, and reliable performance make the HC-05 a popular choice for embedded systems and hobbyist projects.  In summary the HC-05:   * + A versatile Bluetooth module for wireless communication between Arduino and other devices, such as smartphones.   + Supports both master and slave configurations, enabling two-way communication.   + Applications range from remote-controlled robots to data transmission for IoT projects.   These components are critical for real-world applications that require interaction with physical environments.  IM120723009 - HC05 Serial Bluetooth Brick |

|  |  |
| --- | --- |
| **Activity** | |
| **Activity** | **Arduino Project: Smart alarm** |
| In this project we’re going to build an Arduino circuit using the following materials:   * Arduino Nano * PIR Sensor * SR-HC05 (Ultrasonic Sensor) * HC-05 (Bluetooth module)   We are also going to need a stable internet connection, a laptop/computer and an android phone/device. Building an Android Application A logo for a company  Description automatically generatedMIT App Inventor is an intuitive, visual programming environment that allows everyone – even children – to build fully functional apps for Android phones. Those new to MIT App Inventor can have a simple first app up and running in less than 30 minutes. And what's more, our blocks-based tool facilitates the creation of complex, high-impact apps in significantly less time than traditional programming environments. The MIT App Inventor project seeks to democratize software development by empowering all people, especially young people, to move from technology consumption to technology creation. Importing and compiling the app First you need to download the .aia file just click [here](https://drive.google.com/file/d/1jIT5WcmY7ASaGXdVAeByjPlJ76Nuv2Tw/view?usp=sharing) to download the file  After that we need to open the MIT App inventor and login with our google account, after the login you should see something like this:  A screenshot of a computer  Description automatically generated  Now go to project -> Import project (.aia) from my computer…  A screenshot of a computer  Description automatically generated  Now it should appear this, just select choose file and choose the file that you just downloaded it and press ok  A screenshot of a computer  Description automatically generated  Now it will appear this screen:    Go to Build -> Android App (.apk)  A screenshot of a qr code  Description automatically generated  Now just download it by the “Download .apk now” or scan the QR Code over the right side. After that you need to install the .apk file on your android device, every android device is slightly different, but is the same, open the file manager, you should find the .apk file in the downloads folder, click on it and follow the steps. Circuit A circuit board with wires and circles  Description automatically generatedNow we’re going to make the physical part of the project, the Arduino circuit, just follow this diagram. Connections  * Red – 5V * Black – GND * Yellow – D4 * Purple – D5 * Orange – D6 * Green – RX0 * Blue – TX1  CodingA white symbol on a blue circle  Description automatically generatedArduino IDE To code the Arduino, we are going to use the Arduino IDE. The Arduino IDE is an open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board. Code int triggerPin = 5;  int echoPin = 6;  int pirPin = 4;  void setup() {    Serial.begin(9600);    pinMode(triggerPin, OUTPUT);    pinMode(echoPin, INPUT);    pinMode(pirPin, INPUT);  }  void loop() {    digitalWrite(triggerPin, HIGH);    delayMicroseconds(10);    digitalWrite(triggerPin, LOW);    long duration = pulseIn(echoPin, HIGH);    float distance = (duration \* 0.0343) / 2;    int PIR = digitalRead(pirPin);    Serial.println(String(distance) + "/" + String(PIR));    delay(1000);  } How to upload the code After plug-in the Arduino we need to select the correct board, so, go to: Select board and in Search board and type “Arduino Nano” and select it and select the COM Port or /dev/ttyUSB & /dev/ttyACM0  A screenshot of a computer  Description automatically generated  After that just press “Upload” and the IDE will compile and upload into the Arduino nano Board  A screenshot of a computer  Description automatically generated How it works Now you need to power the Arduino nano up, by connecting to a wall outlet with a phone charger or just leave it on the computer plugged in, now turn on Bluetooth and connect to the HC-05, the code should be “1234” or “0000”, after that open the application that we build earlier, it should look like this:    Press “Connect to bluetooth” button and that will show all your Bluetooth paired devices, select the “HC-05”  A screenshot of a computer  Description automatically generated  When you connect it, nothing will appear at first, but when the PIR Sensor detects movement, it Will show a warning sign and the distance between the sensor and the movement object:      To clear the screen you need to press “Close warning” Final Thoughts This document provides a thorough exploration of advanced Arduino concepts, bridging theoretical knowledge with practical applications. From communication protocols like UART, SPI, and I2C to advanced sensors and modules such as the PIR sensor, ultrasonic sensor, and HC-05 Bluetooth module, it equips readers with the tools to design and implement sophisticated projects.  The "Smart Alarm" project exemplifies how these concepts come together, demonstrating the power of Arduino in creating innovative and functional solutions. With clear instructions and user-friendly tools like the MIT App Inventor, this guide makes advanced technology accessible to learners and hobbyists alike.  Overall, this resource highlights the educational value of Arduino, fostering creativity and hands-on learning. It serves as both a foundation and inspiration for those looking to explore and expand their capabilities in electronics and programming. |
| **Sources:** |
| **Extra contents:** |

|  |
| --- |
| **Activity (Assessment of Module 4.2)** |
| **Type**: Project-based assessment |
| Based on the knowledge acquired in the training, participants should develop a small functional project using Arduino and sensors to their liking, which can first be simulated in Tinkercad before physical implementation.  Assessment criteria:  Originality of the idea (10%)  Degree of difficulty (20%)  Correct functioning of the circuit (30%)  Well-structured and functional code (30%)  Presentation and explanation of the project (10%) |

1. [↑](#footnote-ref-1)