Erasmus+

2Play2Learn

Reviving hands-on educational play for learning skills of tomorrow

PROJECT N° 2019-1-UK01-KA201-061466

MODULE 2

Scratch 2.0 – DIY Electronic Kits

DEVELOPED BY IDEC & PLATON

CIVIC idec ΕΚΠΑΙΔΕΥΤΗΡΙΑ ΠΛΑΤΩΝ dlearn scholé Emphasys CENTRE CCS Digital Education

MODULE 2 – Scratch 2.0

MODULE DESCRIPTION

Scratch is a visual programming tool which allows the user to create animations and games with a drag-and-drop interface. It allows you to create your own computer games, interactive stories, and animations using some programming techniques without actually having to write code. It’s a great way to get started programming on the Raspberry Pi with young people.

The version of Scratch included with the Raspberry Pi has a number of unique features; one of the most useful is its ability to communicate with the GPIO pins (General Purpose Input Output). These pins allow you to connect your Raspberry Pi to a range of devices, from lights and motors to buttons and sensors.

Co-funded by the
Erasmus+ Programme
of the European Union

DIY Exercise 1 – Blinky LED

We will now put together a small demo using the Pi and some components. In this simple demo we will use Scratch to program the Pi to blink an LED.

First, let's wire things up.

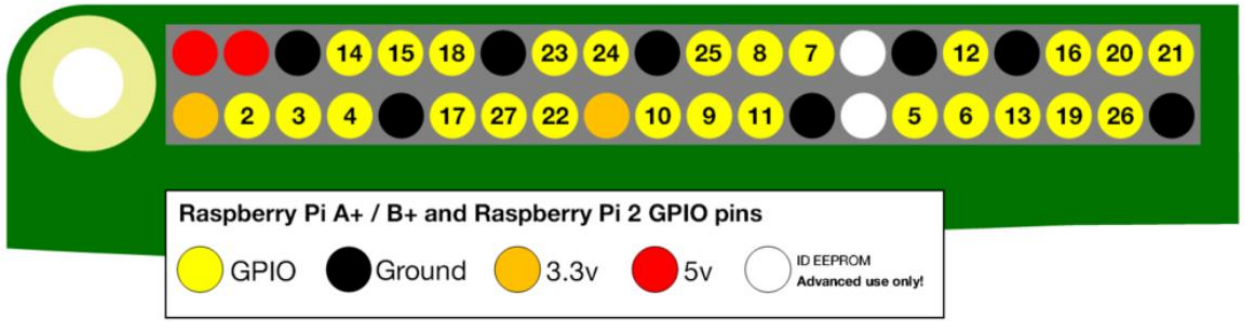
Co-funded by the Erasmus+ Programme of the European Union

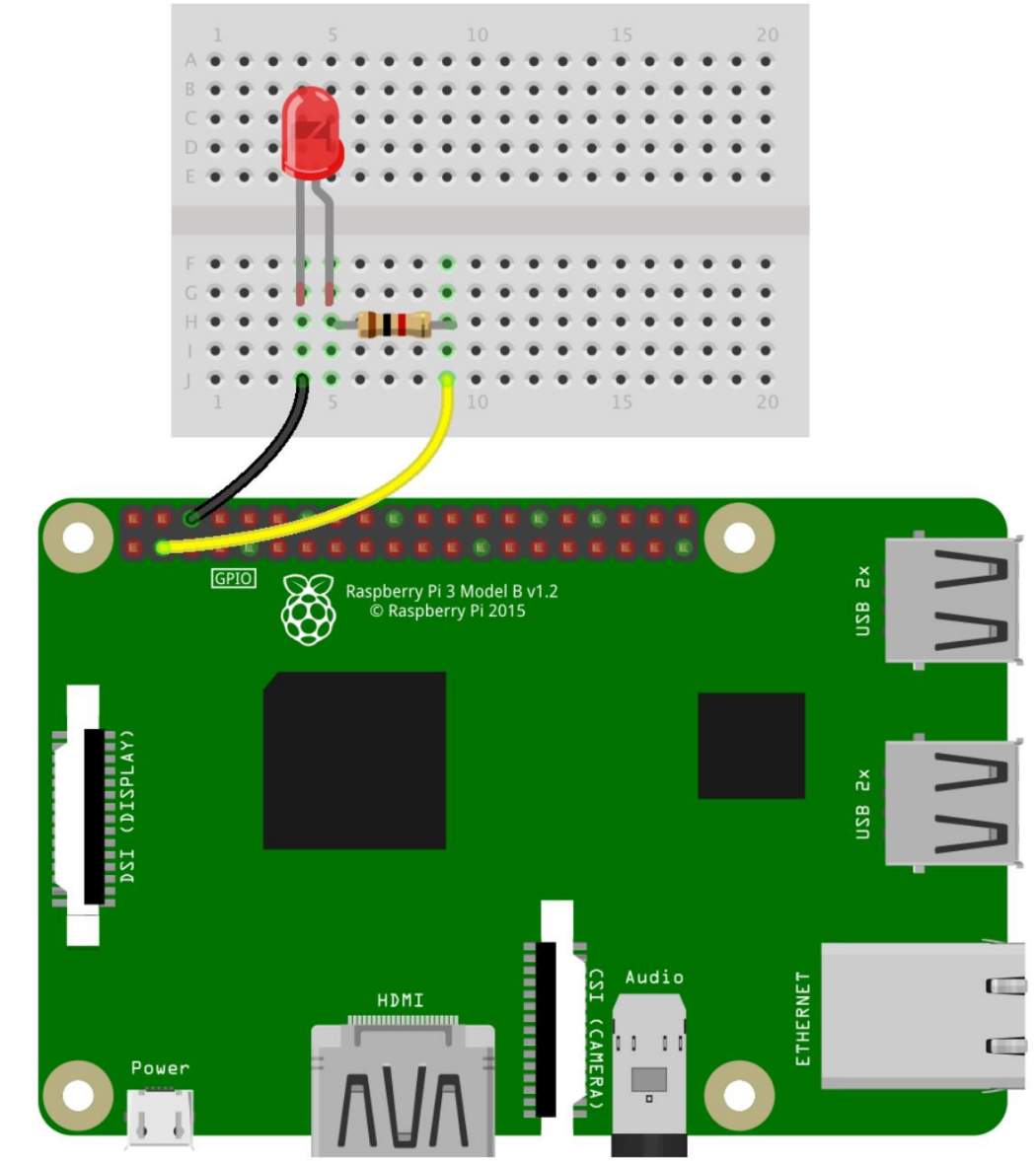
MODULE 2 – Scratch 2.0

DIY Exercise 1 – Blinky LED

You will need handy:

- one led (any color)
- one resistor that is 1K or less
- 2 female to male jumper wires



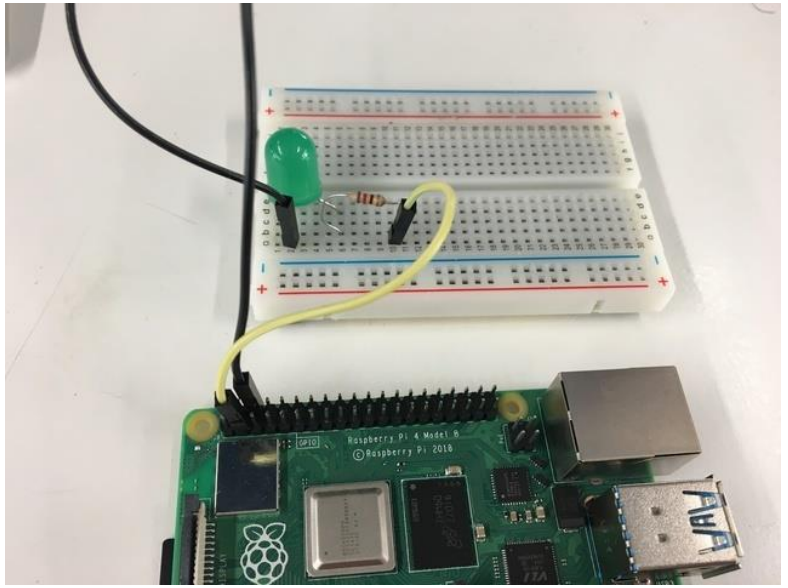


Co-funded by the
Erasmus+ Programme
of the European Union

MODULE 2 – Scratch 2.0

DIY Exercise 1 – Blinky LED

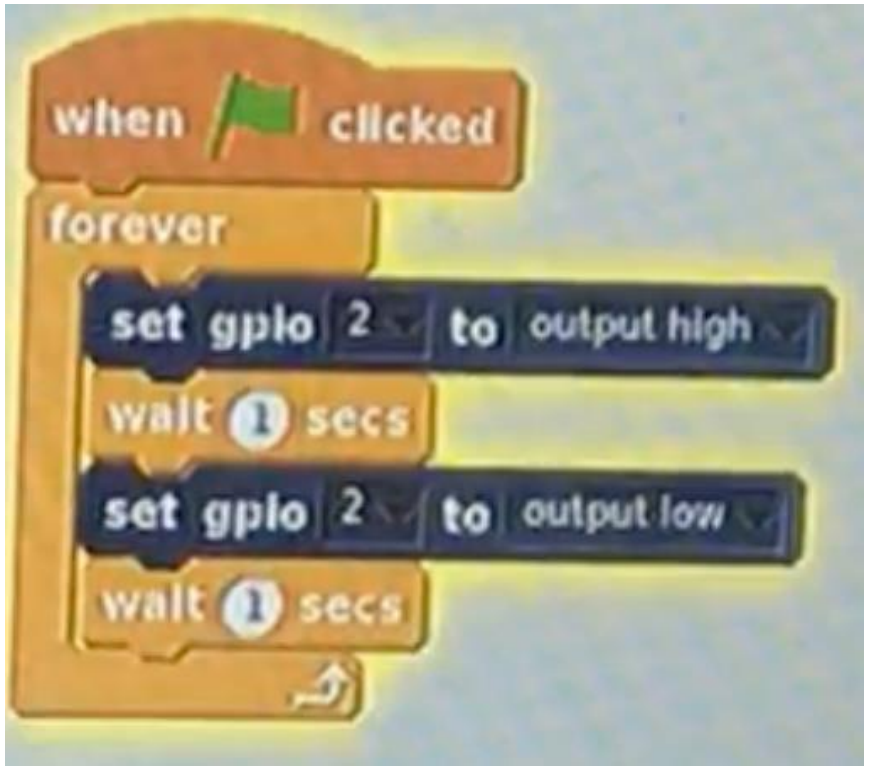
- Connect one female to male jumper wire from GPIO pin 2 on the pi to any spot on a bread board
- Next, connect the resistor (1K or less will do) from the GPIO 2 jumper wire to another spot on the breadboard
- Now take the LED and determine which leg is longer, and connect that leg to the other end of the resistor with the opposite end going to another spot on the bread board.
- Lastly, connect male to female jumper wire from the unconnected end of the led to a ground pin on the Pi.



Co-funded by the Erasmus+ Programme of the European Union

DIY Exercise 1 – Blinky LED

Next, we'll program the Pi in scratch to blink the LED.



Co-funded by the
Erasmus+ Programme
of the European Union

Your LED is **BLINKING**

Co-funded by the
Erasmus+ Programme
of the European Union

DIY Exercise 2 – Creating a buzzer

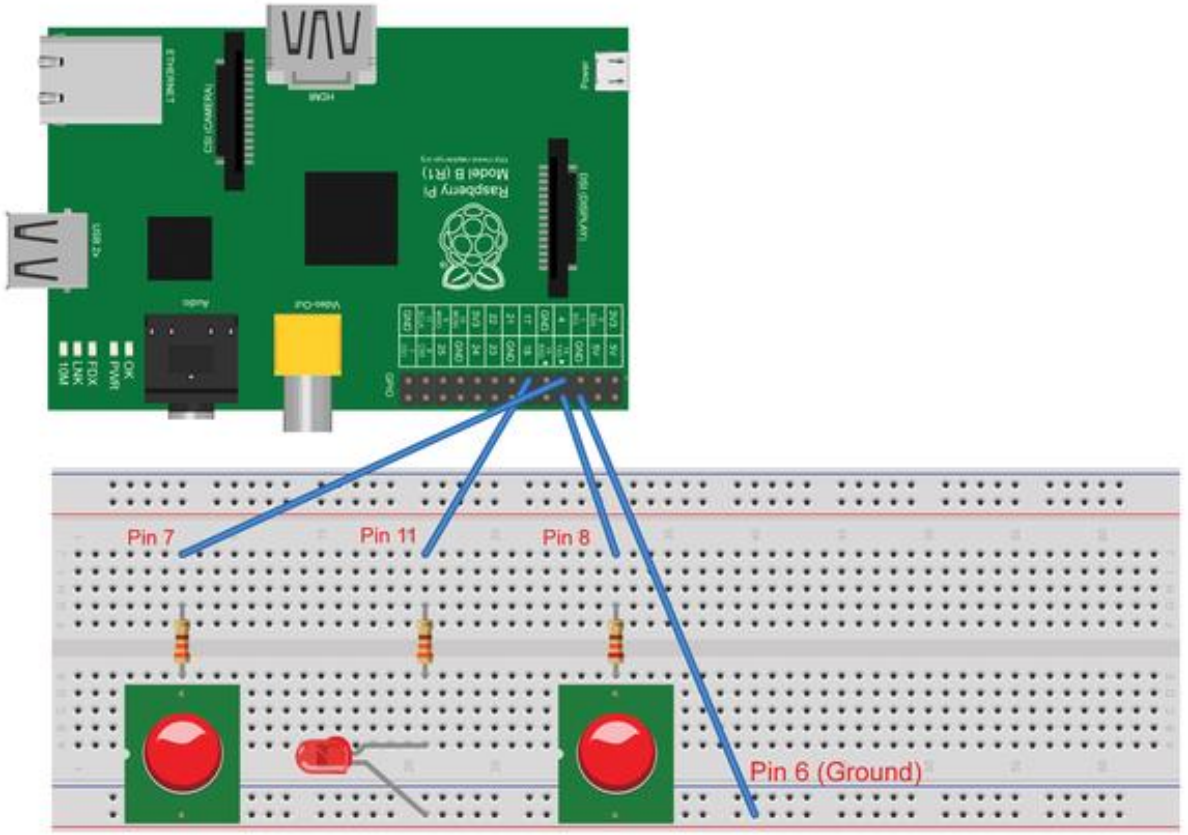
We will now put together a small demo using the Pi and some components. In this simple demo we will use Scratch to program the Pi to create a multiplayer quick-reaction game.

To see which player has the faster reaction, you need to wire two buzzers to the breadboard. The Scratch program will control when the players are allowed to press the button, detect the first buzzer press, and keep score.

Co-funded by the Erasmus+ Programme of the European Union

DIY Exercise 2 – Creating a buzzer

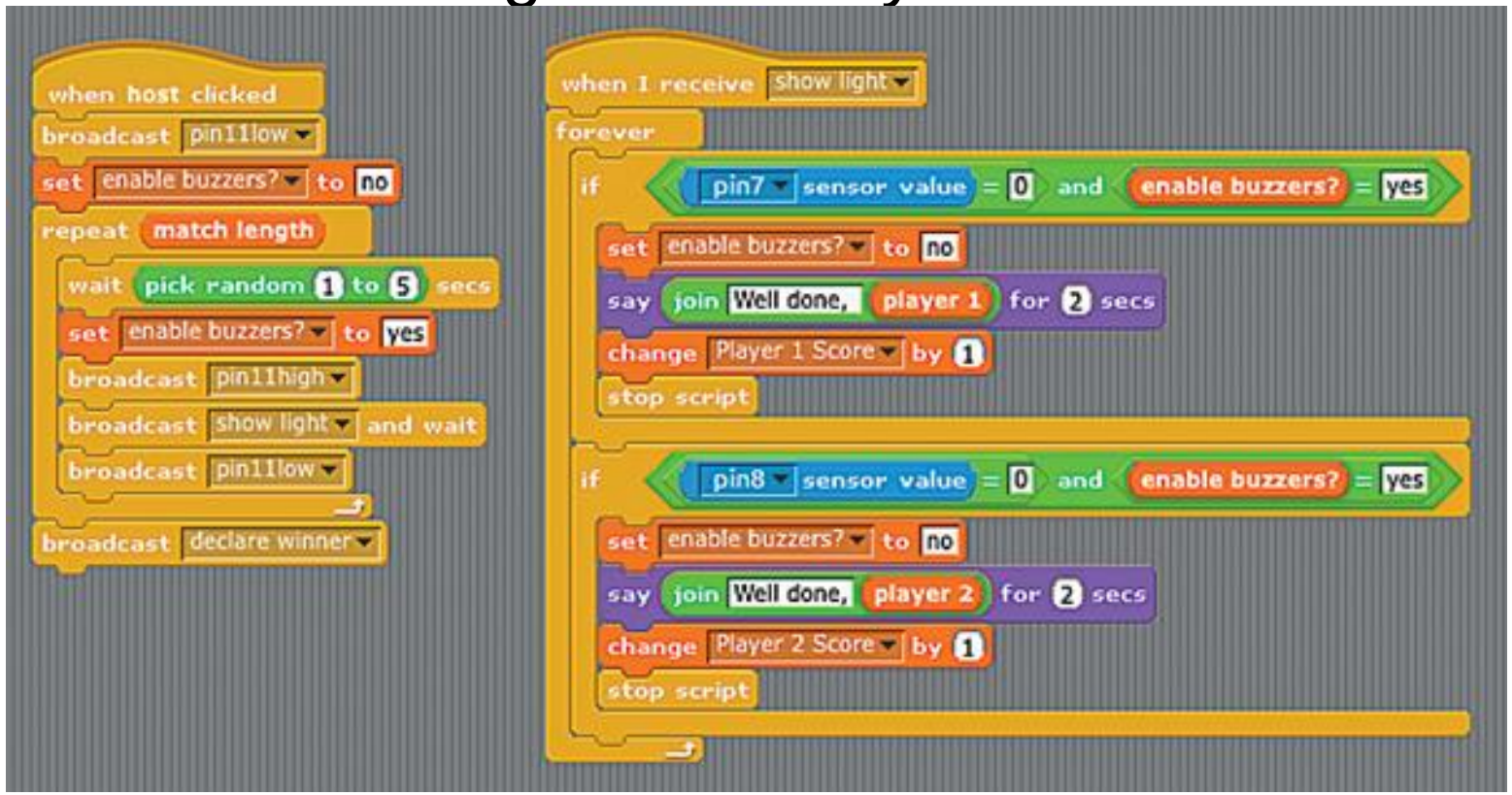
For the buzzer, I'll use two simple push-button switches, one for each of the two players. Player one's buzzer will connect to pin 7, and player two’ s will connect to pin 8. (See figure on the right for the buzzer circuits.) To detect the buzzer clicks, the Scratch script needs to monitor the input on the Raspberry Pi's pins. ScratchGPIO treats pins 3, 5, 7, 8, 10, 19, 21, 22, 23, 24, and 26 as input.



Co-funded by the
Erasmus+ Programme
of the European Union

DIY Exercise 2 – Creating a buzzer

The two scripts show the main logic for the game. The script that begins with the **when host clicked** block starts the game. I added a new variable called **enable buzzers?** and assigned it the value **no**, to prevent players from buzzing in too early.



Co-funded by the Erasmus+ Programme of the European Union

DIY Exercise 2 – Creating a buzzer

The **repeat()** loop uses a variable called **match length** to control how many matches or sets each game will have. This approach allows the players to achieve the best two out of three, for example. This setup can be controlled by a slider control on the stage.

After waiting a random amount of time from **1** to **5** seconds, the script enables the buzzers and then turns on the LED by broadcasting a **pin11high** message. When the LED lights up, that's the signal for the players to buzz in. The **broadcast(show light) and wait** block coordinates the programming needed to detect the players' buzzers.

Co-funded by the Erasmus+ Programme of the European Union

The **when I receive(show light)** script, detects the pin input. The **forever** loop ensures that the script will continue running until either player one or player two clicks a buzzer. The **()sensor** value block has drop-down options for the ScratchGPIO inputs.

Additionally, the first **if()** block checks whether pin 7 (player 1) is on and whether the **enable buzzer?** variable is equal to **yes**. If both conditions are true, the script increments player one's score, disables the buzzer, and stops the current script. The second **if()** block performs the same check for pin 8 (player 2).

Co-funded by the Erasmus+ Programme of the European Union

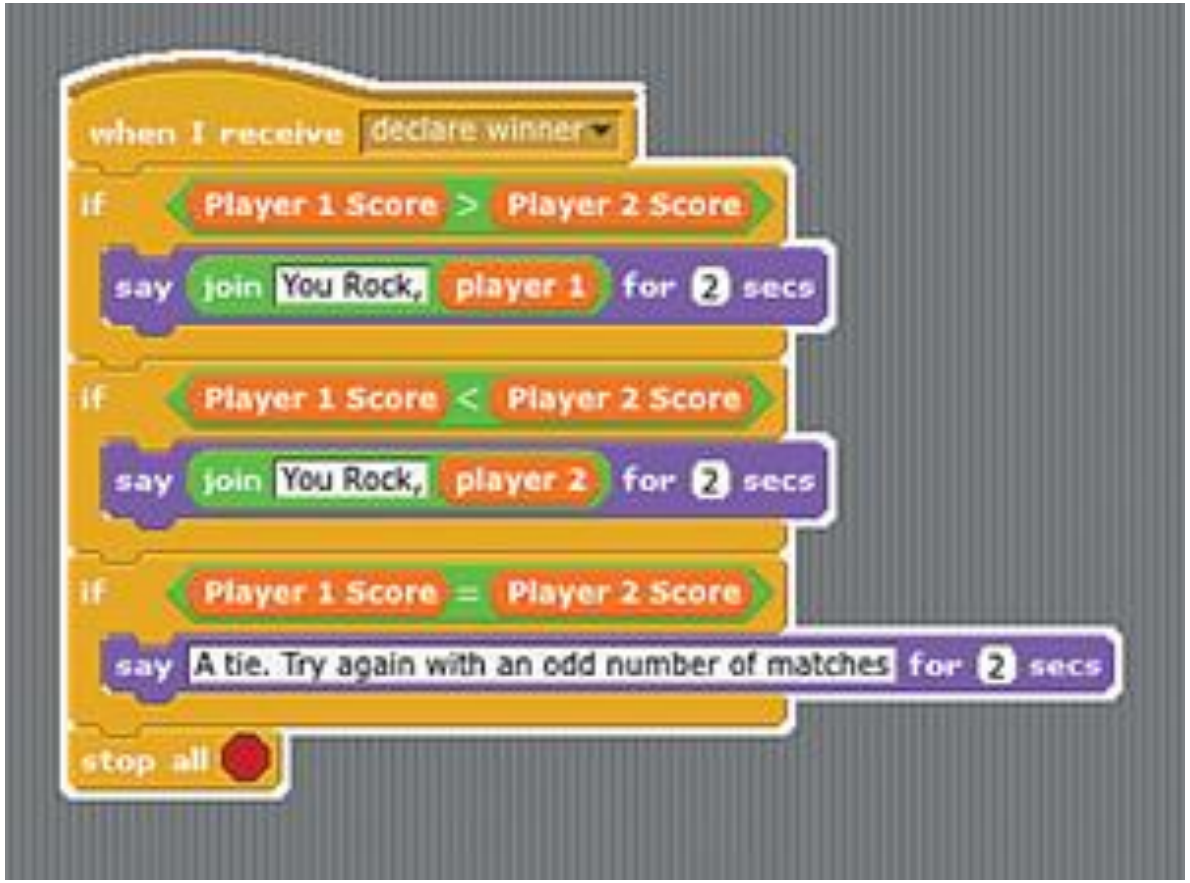
DIY Exercise 2 – Creating a buzzer

The input pins have a default value of **1** in the "off" position. When input is detected, such as clicking the buzzer, the value of the pin becomes **0**. You can see this by enabling the **pin7** sensor value in the stage monitor.

Disabling the buzzer prevents the other player from buzzing in and getting a point. Stopping the script breaks the **forever** loop and signals the **when host clicked** script to pick up where it left off, which will be to run the **broadcast(pin11low)** block to turn off the LED. The advantage of using the **broadcast(show light) and wait** block is that the script will pause until all the **when I receive(show light)** scripts complete.

Co-funded by the Erasmus+ Programme of the European Union

Then, the script continues to randomly light the LED for the specified number of matches. At the end of the game, the program declares a winner with the **broadcast(declare winner)** block at the end of the **when host clicked script**.



The **when I receive(declare winner)** script determines which player has the highest score and then announces a winner.

Co-funded by the Erasmus+ Programme of the European Union

LET'S PLAY

Please don't break the buttons…

Co-funded by the
Erasmus+ Programme
of the European Union

References

- https://learn.adafruit.com/programming-with-scratch-on-raspberry-pi/demo
- https://www.raspberry-pi-geek.com/Archive/2014/07/Creating-a-multiplayer-quick-reaction-game/(offset)/2

Co-funded by the Erasmus+ Programme of the European Union

Extra resources

- https://www.instructables.com/id/Physical-Computing-Scratch-20-for-Raspberry-Pi/

Co-funded by the
Erasmus+ Programme
of the European Union

Erasmus+

Reviving hands-on educational play for learning skills of tomorrow

PROJECT N° 2019-1-UK01-KA201-061466

CIVIC idec ΕΚΠΑΙΔΕΥΤΗΡΙΑ ΠΛΑΤΩΝ dlearn scholé Emphasys CENTRE CCS Digital Education