PROGRAMMING RESOURCES - ACTIVITY SHEET 13

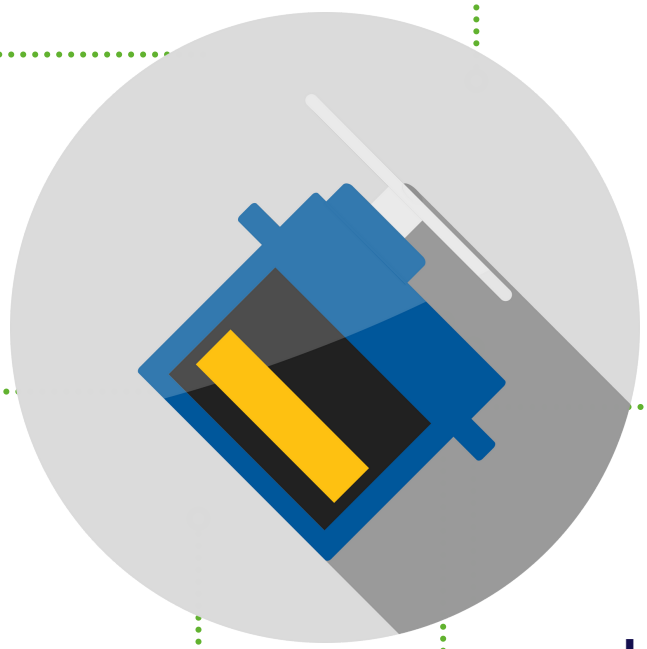
SERVOS MAKE THINGS MOVE!

#R1AS13

What is it?

Servo is a driver for keeping the position. It is suitable to control a system with constant angle change and can remain its status.

Available on



Pre-requisites

- R1AS03 - Buttons and LED Display

Duration

25 minutes

Level of difficulty

Intermediate

Material

- 1 Programming board "***STM32 IoT Node Board***"
- Micro-B USB Cable
- 1 SG-90 Mini Servo(1.6kg)
- Jumper Wires

LEARNING OBJECTIVES

- Set an object in motion

This activity sheet is part of the Let's STEAM project funded with support from the European Commission through the Erasmus + Strategic Partnership programme. Its content reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

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

SERVOS MAKE THINGS MOVE!

A Servo is a motor with a set of automatic control systems, which consists of an ordinary **DC motor** (rotary electrical motors that converts direct current electrical energy into mechanical energy), a reduction gear unit, a **potentiometer** (voltage divider used for measuring electric potential or voltage) and a control circuit. It can define the rotation angle of the output shaft by sending signals. Usually, a servo has a maximum rotation angle (e.g. 180 degrees).
Resources: https://en.wikipedia.org/wiki/DC_motor, https://en.wikipedia.org/wiki/Potentiometer

The servo system can be controlled by impulse, which can change its width. We use a control cable to transmit the impulse. The cycle of a servo reference signal is 20ms and the width is 1.5ms. The position defined by the servo reference signal is the middle position. Since servo has a maximum rotation angle, the definition of middle position is from this position where the maximum value and the minimum value are the same.

STEP 1 - MAKE IT

1 Connect the servo to the board

There are many ways to wire a servo to your board. You can use any analog output pin (PWM pins) to connect the control pin. In our example, we will use the **D4** pin. The servo will be connectect as follows:

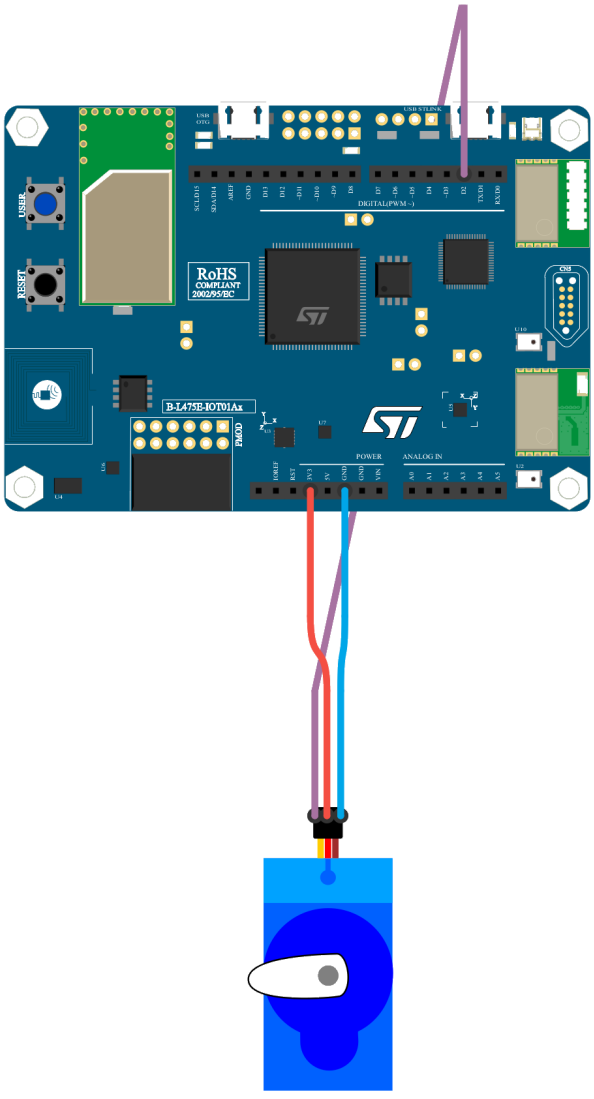
- Black for **GND**
- Red for **V+** (**3V3**)
- Orange for **SIG** (**D4**)

2 Connect the board to the computer

With your USB Cable, connect the board to your computer by using the **USB ST-LINK connector** (on the right corner of the board). If everything is going well you should see a new drive on your computer called **DIS_L4IOT**. This drive is used to program the board just by copying a binary file.

3 Open MakeCode and create a new blank project

Go to the **Let's STEAM MakeCode editor**. On the home page, create a new project by clicking on the "New Project" button. Give a name to your project more expressive than "Untitled" and launch your editor.
Resource: makecode.lets-steam.eu



Connect the servo to the board

SERVOS MAKE THINGS MOVE!

STEP 1 - MAKE IT

After creating your new project, you will get the default "ready to go" screen shown here.

Program your board

4

Inside the MakeCode Javascript Editor, copy/paste the code available in the **Code It Section** below.

Before trying this program on the board, you can try it directly inside the simulator. If you change the values 0 and 180, you will see the result directly.

If not already done, think of giving a name to your project and click on the "**Download**" button. Copy the Binary file on the drive **DIS_L4IOT**, wait until the board finish blinking and your servo will start to move!

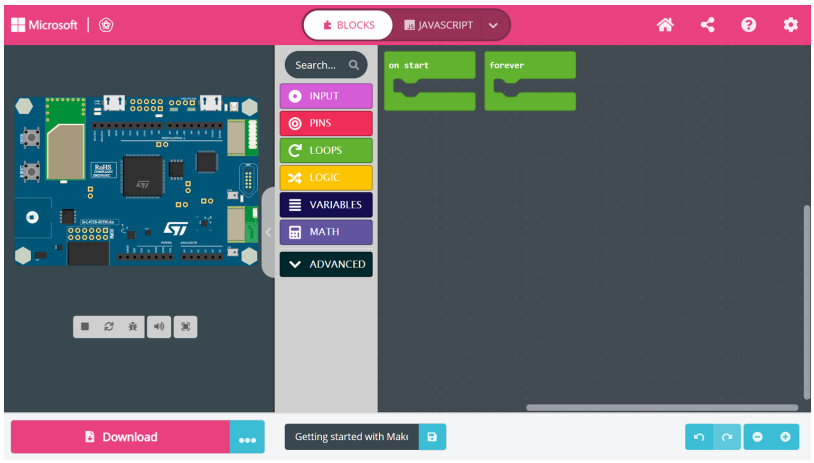
Run, modify, play

5

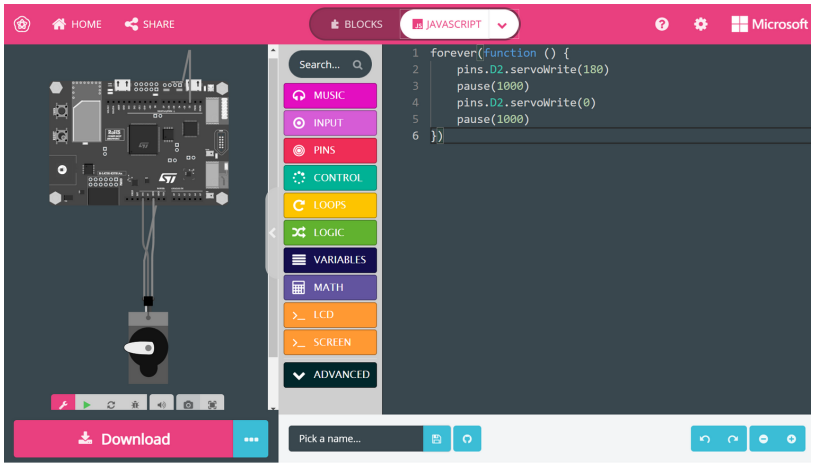
Your program will automatically run each time you save it or reset your board (push the button labelled RESET).

If everything is working well, your servo will start to move.

Try to understand the example and start modifying it by changing the period between the two moves.



Default "ready to go" screen



Makecode Javascript Editor

Your servo starts to move

STEP 2 - CODE IT

```
// goes from 0 degrees to 180 degrees
forever(function () {
   // tell servo to go to position 180 degree
   pins.D4.servoWrite(180)
   // wait for the servo to reach the position
   pause(1000)
   // tell servo to go to position 0 degree
   pins.D4.servoWrite(0)
   // wait for the servo to reach the position
   pause(1000)
})
```

How does it work?

This sample is pretty straightforward as it is the classical "blinky" adapted to a servo.

The main instruction is pins.D2.servoWrite(XXX). This instruction asks the servo to rotate at an angle of XXX degrees (as set by your specific needs depending on the project you are developing).

To move between two positions, the servo takes some time so we always need to add a delay before starting another move.

This program just sweeps left and right forever!

Compared with an ordinary DC motor, a servo rotates within a certain angle range only, while an ordinary DC motor rotates in a circle.

A servo cannot rotate in a circle. An ordinary DC motor cannot give us feedback about rotation angle but a servo can do it. Their usages are hence different.

Ordinary DC motors use a whole circle rotation as power while servo uses a certain angle of an object it controlled such as a robot joint.

SERVOS MAKE THINGS MOVE!

STEP 3 - IMPROVE IT

1. Try to **reduce as much as possible the value of the pause** to remove any motion stop.

2. Add instructions to **make a short stop in the middle position**. Adapt the delay of the pause to be sure that the stop was very short.

3. Transform this program to **make a timer with a servo**. At each step, move the servo of 3 degree. Adapt the delay such as each step take approximatly 1s.

4. Start the **sweep move** only when the USER button was clicked.

GOING FURTHER

Servomotor - Learn more about the mechanism and control operation of servomotor. *https://en.wikipedia.org/wiki/Servomotor*

Servo Motors with micro:bit - All about buttons and their use in MakeCode with Shawn Hymel, Technical Content Creator. *https://www.youtube.com/watch?v=okxooamdAP4&t=200s, https://shawnhymel.com*

DIY Color Sorting Robotic Arm - Learn how to make your own DIY color sorting robotic arm using ultrasonic and IR sensors. *https://thestempedia.com/project/diy-color-sorting-robotic-arm/*

Explore other activity sheets

R1AS14 - Create an egg timer