Introducción a la programación de microcontroladores con Arduino

Grupo:

- Mauro Bosetti
- Thyago Santos
- Pablo Spizzamiglio

13 de julio de 2023

INITIATIVE TRACKER

DESCRIPCION DEL PROYECTO

Decidimos elaborar nuestra propia propuesta de proyecto en lugar de una encontrada en internet. Esto nos permitio explorar el proceso de partir de una necesidad y llegar a una solucion con lo que habíamos aprendido en la materia. Para entender de qué se trata, es necesario conocer un poco del funcionamiento de *Dungeons & Dragons,* el conocido juego de rol de mesa.

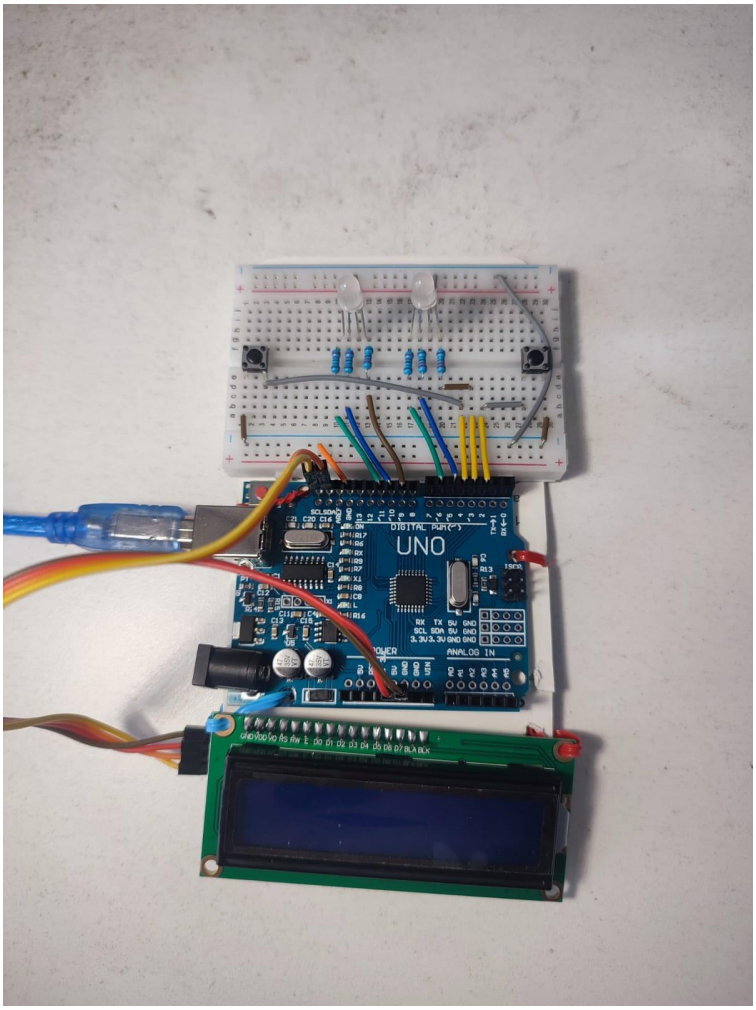
CONTEXTO

En el juego, el orden de los turnos siempre es variable. Esto quiere decir que, en un momento determinado, el orden de los turnos puede ser [Jugador A, Jugador B, Jugador C], y una hora más tarde puede terminar siendo [Jugador C, Jugador A, Jugador B], dependiendo de tiradas de dado.

Los juegos de rol no vienen en una caja o con un kit de materiales particular, por lo que la gran mayoria del tiempo ningun elemento del juego sirve para recordar de quién es el turno. Ademas de esto, el orden se determina utilizando un dado de 20 caras, por lo que cada jugador puede tener asignado un número de turno del 1 al 20. El proceso de definir el orden, entonces, suele ser engorroso: cada jugador va cantando su número (el cual llamamos iniciativa), mientras que alguien va haciendo lo posible por mantener una lista ordenada de estos valores.

Nuestra idea de proyecto busca facilitar el proceso de cantar tu número, mantenerlo ordenado, y tener claro de quién es el siguiente turno.

¿QUÉ HAREMOS?



Utilizando un display LCD y dos luces LED RGB, vamos a armar un tracker de iniciativa, para siempre saber de quién es el turno.

Utilizaremos un par de botones para avanzar y retroceder turnos, y colores asignados a cada jugador para que los dos LEDs RGB indiquen no solo de quien es el turno, sino ademas de quien es el turno siguiente. Para mantener el estado y ademas permitir el ingreso comodo de datos, levantamos un servidor Node que se conectara al Arduino mediante el puerto serie usando el firmware “StandardFirmata”, y ademas una web como interfaz visual para ingresar datos.

MATERIALES

- Arduino UNO o equivalente
- 2 LEDs RGB ánodo común
- Display LCD 16x2 con módulo I2C
- 6 resistencias de 220 ohms

- 2 botones
- 1 protoboard pequeño
- Cables

CIRCUITO

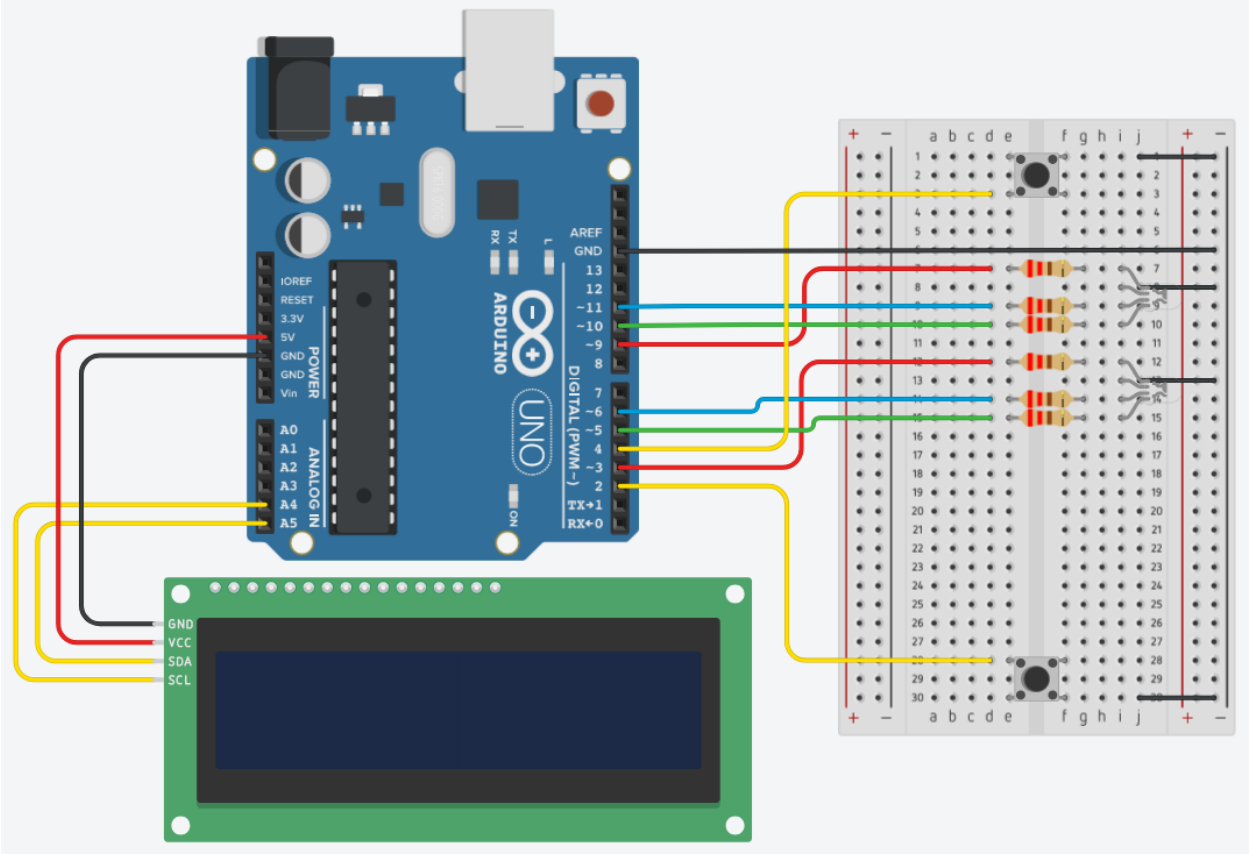
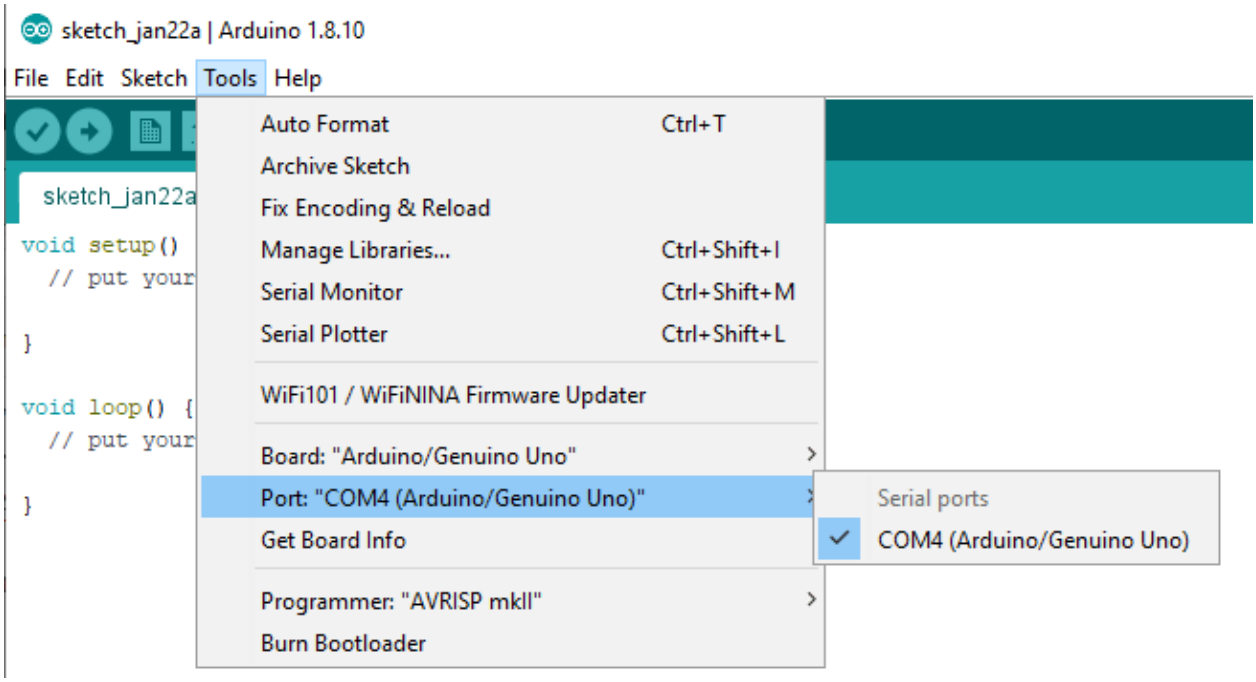


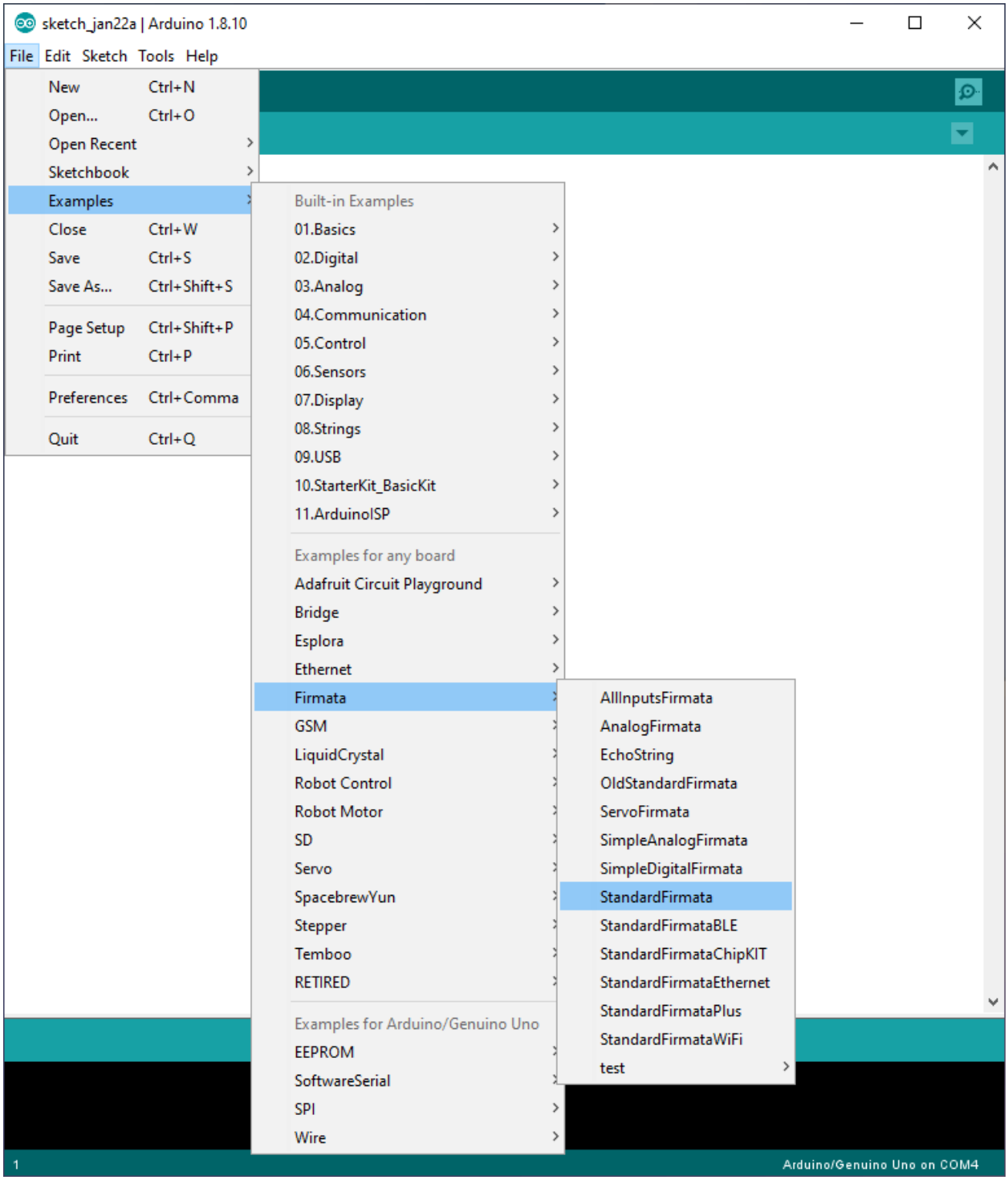
Figura 1 - Circuito

REQUISITOS PARA EJECUTAR EL SOFTWARE

- Instalar Arduino IDE 2[1]

- Cargar el Arduino con el firmware “StandardFirmata” (utilizamos la version que viene incluida con Arduino IDE 2)





- Instalar Node 18
- Instalar el compilador de C++ (requerido por la librería serialport para Node)

    - Ubuntu: sudo apt install g++
    - Fedora: sudo dnf install gcc-c++
- Clonar el repositorio https://github.com/amuroBosetti/initiativeTracker
- Instalar dependencias con npm install
- Levantar el servidor con npm start

En el puerto 3000 estará corriendo el frontend, con el cual se puede acceder a las vistas /player para los jugadores y /admin para quien esté dirigiendo el juego.

DESCRIPCION DE LA APLICACIÓN

La aplicación consiste en un backend hecho con Node y Express, que se comunica con el Arduino mediante el puerto serie utilizando el protocolo de comunicacion genérico Firmata, mediante la libreria Johnny-Five. Y un frontend hecho con React, el cual posee dos vistas, la del jugador (fig. 2), que permite a los participantes agregar nuevos personajes desde sus teléfonos con un valor de iniciativa y un color, y la vista del administrador (fig. 3), que permite ver a todos los personajes existentes, editarlos, borrarlos y agregar nuevos.

Figura 2 - Vista del jugador

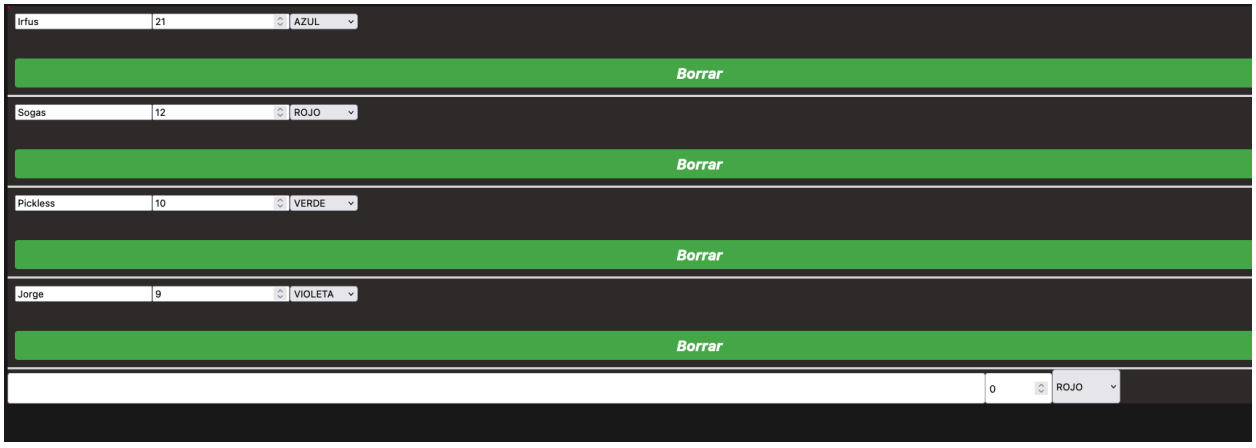


Figura 3 - Vista del administrador

Este frontend se comunica mediante HTTP con el servidor, haciendo operaciones sobre la lista de jugadores. El servidor mantiene en memoria este estado y, cada vez que se modifica, refresca el estado de los RGBs y el display.

PROBLEMAS ENCONTRADOS

Durante el desarrollo encontramos varios problemas, en su mayoria originados por no tener un claro instructivo paso a paso sino una idea propia.

Uno de ellos fue tener que recortar el alcance del producto: originalmente ideamos al menos tres luces RGB, e incluso pensamos en la posibilidad de tener redundancia de luces, dos para cada lugar de la fila, para mayor efecto visual. Sin embargo, nos vimos limitados por la cantidad de pines que permiten la modulación por ancho de pulsos (del inglés pulse-width modulation, PWM) ya que cada LED RGB ocupa tres pines, uno para cada color, y el Arduino UNO sólo tiene seis.

Otro problema surgió durante la instalación de la librería Johnny-Five[2], la cual depende a su vez de la librería SerialPort[3]. Esta última precisa compilar los bindings de C++[4] para poder interactuar con los puertos serie provistos por el sistema anfitrión, para ello utiliza node-gyp[5] que a su vez tiene como prerrequisito que el compilador de C++ esté disponible en el sistema.

REFERENCIAS

[1] https://docs.arduino.cc/software/ide-v2

[2] https://johnny-five.io/

[3] https://serialport.io/

[4] https://serialport.io/docs/api-bindings-cpp

[5] https://github.com/nodejs/node-gyp