Memoria 8Bit bag

DISEÑO DE SISTEMAS EMPOTRADOS – GIC 2019/2020

GRUPO 5 - YAROSLAV DYTKO

Contenido

Introducción

El proyecto trata de construir un dispositivo luminoso basándose en una mochila y una matriz de diodos led RGB, todo ello controlado por una placa ESP32 que proporcionara comunicación mediante la interfaz Bluetooth con un dispositivo móvil Android mediante el cual se podrá controlar la matriz led para que realice diferentes funcionalidades. El proceso de construcción de este dispositivo consta de varias fases de producción/trabajo que entienden tanto la parte de programación como la parte de trabajo con el material hardware, l una vez finalizadas todas las fases darán como resultado el dispositivo terminado.

El proceso de construcción consistirá en las siguientes fases:

- Creación de matriz led, montaje de conexiones, montaje de circuito.
- Programación de aplicación de control para la plataforma Android.
- Programación del firmware para la placa controladora ESP32 Dev Board.

En esta fase de introducción fijare una serie de objetivos generales por cumplir:

- Creación de la base hardware para el proyecto – consiste en crear y montar toda la parte hardware relacionada con el proyecto.
- Programación de aplicación móvil para controlar el dispositivo – consiste en crear la aplicación que permitirá controlar el dispositivo de manera cómoda y sencilla.
- Programación del firmware para la placa controladora – la placa controladora se encargará de enviar señales a la matriz de leds y reproducir los efectos visuales programados previamente.

Aparte de los objetivos generales que abarcan el desarrollo del proyecto en general también fijare una serie de objetivos más concretos por cumplir, estos tienen el mismo grado de importancia que los objetivos generales:

- Establecer conexión entre el dispositivo móvil (teléfono móvil) y la placa controladora por medio de protocolo de comunicación Bluetooth.
- Implementar una lógica de control para la aplicación del teléfono móvil.
- Enviar datos desde el teléfono móvil a la placa controladora.
- Desarrollar un protocolo de comandos y parámetros de control para la placa controladora.
- Implementar un método de recepción e interpretación de comandos de control en la placa controladora.
- Montaje y conexionado de la matriz led.
- Implementar los efectos luminosos que serán visualizados posteriormente.
- Visualizar efectos luminosos con la matriz de leds y placa controladora.
- Mantener el coste del proyecto lo más bajo posible.

Materiales

Para el desarrollo del proyecto necesitare los siguientes materiales:

Tira de diodos LED RGB 5V, 5m 60leds/m IP30
ESP32 Dev Board
Plancha de gomaespuma 40cm x 60cm x 5mm – 1 unidad
Condensador electrolítico 6.3v 470mF o 16v 470mF – 1 unidad
Conector USB macho tipo A – 1 unidad
Cubierta encuadernación DIN A4 0.45mm plástico – 2 unidades
Cinta adhesiva doble cara 3M 3metros – 2 rollos
Cable – 1.5 metros
Resistencia 200 – 400 Ω - 1 unidad
Mochila – 1 unidad

Esquema principal

El esquema principal del dispositivo(Ilustración 1) es bastante simple e intuitivo, básicamente se usan tres cables, tanto la placa controladora como la matriz led se alimentan de la misma fuente de alimentación por vía de conector USB que se puede ver en el esquema, salvo que a la hora de conectar la placa a la alimentación hay que incluir en la línea un condensador electrolítico tal como se muestra en el esquema. También a la hora de conectar la línea de datos desde la placa hasta la matriz hay que incluir una resistencia en dicha línea para proteger la placa de los saltos de intensidad si se produjesen sobre dicha línea.

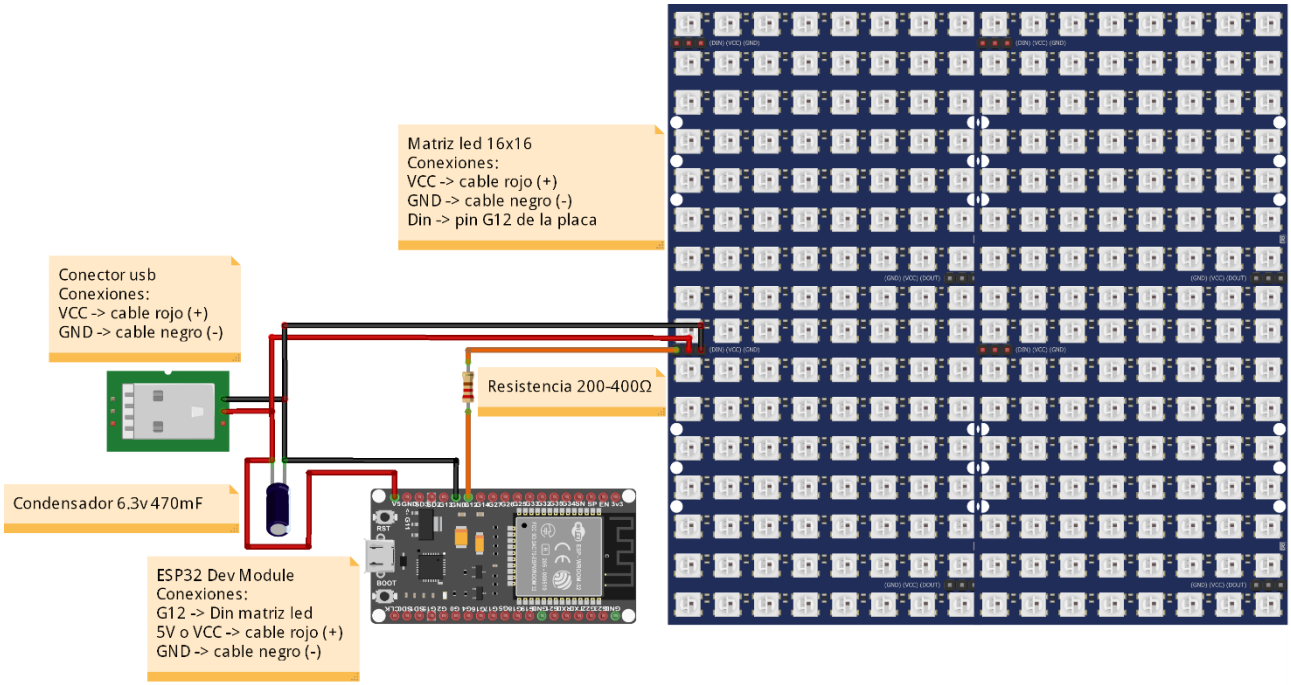


Ilustración 1. Esquema principal del dispositivo

Problemas con el circuito principal: inestabilidad de funcionamiento de la placa.

Construcción de matriz led

Para construir la matriz de los leds lo primero que he hecho es recortar las base donde se irán pegando las tiras, las dimensiones de la base son 27 x 27 centímetros, hay que usar algún tipo de material ligero por cuestiones de peso y rígido para darle robustez a la estructura y que no se deforme fácilmente. He utilizado la gomaespuma de 5 mm de grosor, color negro. Recomiendo recortar dos piezas iguales ya que una será la base y otra será el marco que se fabricara los pasos posteriores y es del mismo material.

Una vez recortada la base hay que cortar la tira de los dodos led en 16 tiras más pequeñas, cada una de 16 diodos led de largo, lo que nos da en total 256 diodos led.

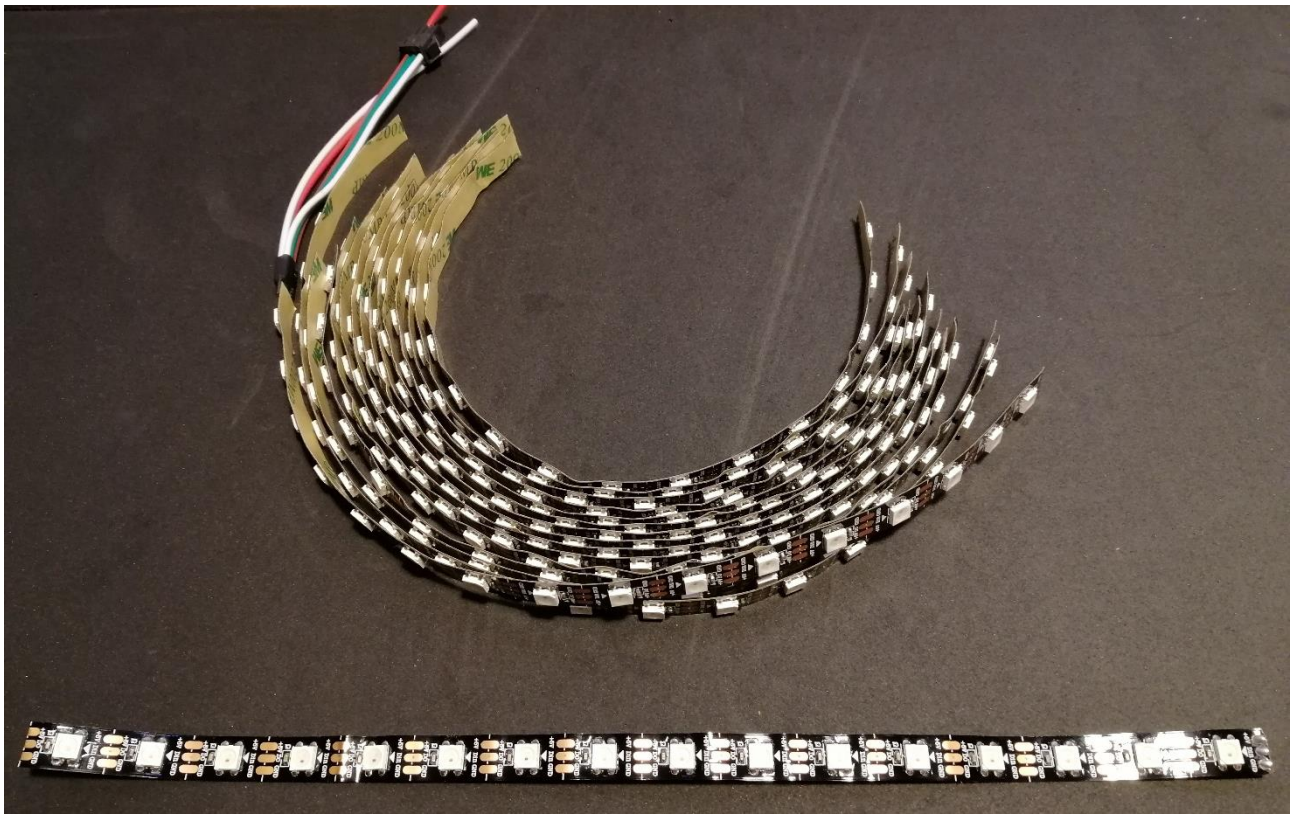


Ilustración 2. Tiras led cortadas

Después de cortar las tiras led hay que pegarlas a la base de gomaespuma empezando por abajo, la primera fila hay que pegarla dejando aproximadamente 5-6mm de margen desde la base de la pieza, las demás tiras hay que pegarlas con el margen de 16.666 mm entre la base de la tira inferior y base de la superior (ver Ilustración 3).

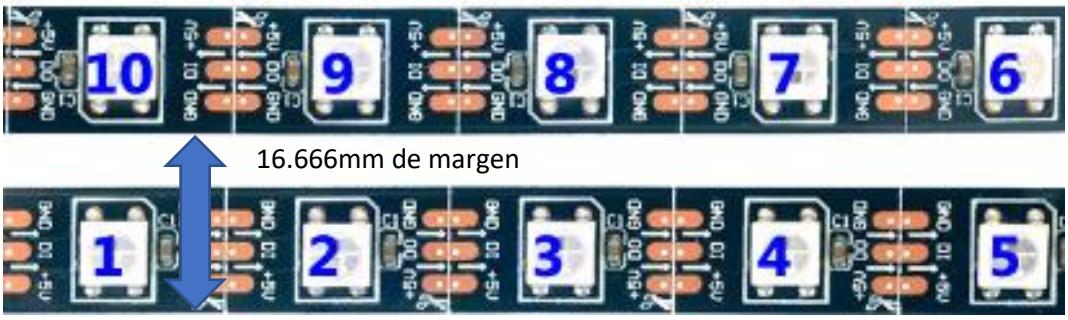


Ilustración 3. Margen entre tiras

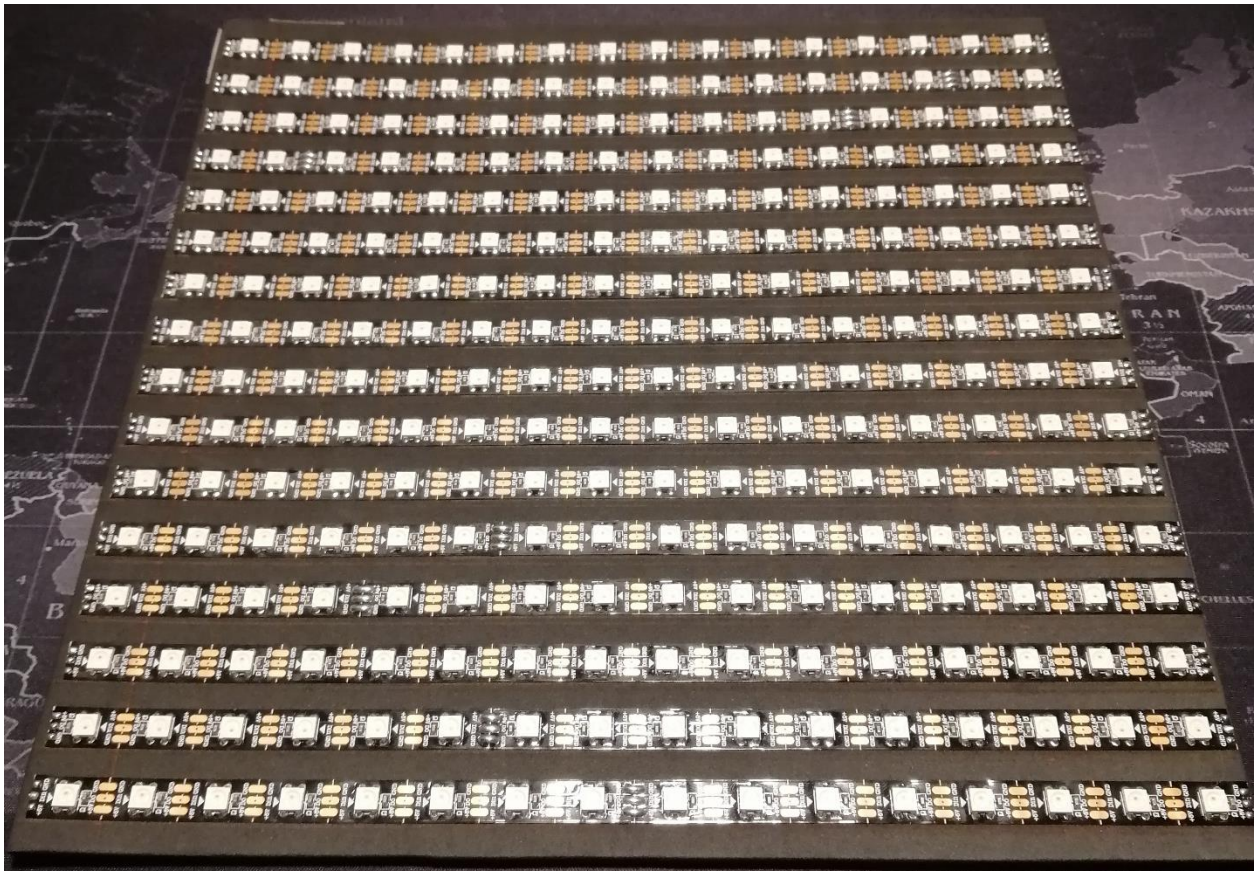
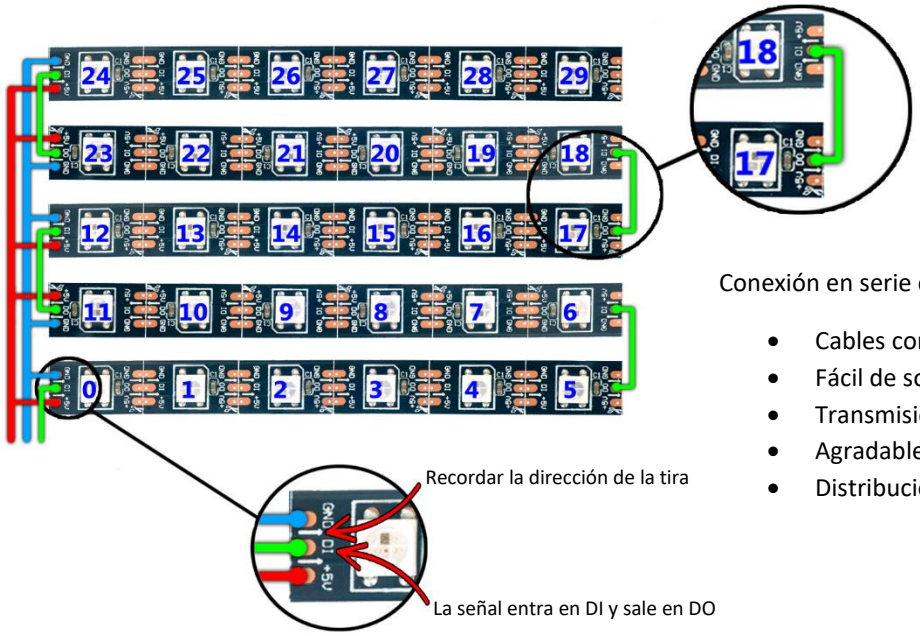


Ilustración 4. Resultado tras pegar las tiras led

Una vez pegadas las tiras led se pueden hacer las soldaduras para interconectarlas todas, voy a seguir el siguiente esquema para hacer las conexiones:



Conexión en serie con alimentación en paralelo

- Cables cortos
- Fácil de soldar
- Transmisión de señal segura
- Agradable a la vista
- Distribución de alimentación optima

Ilustración 5. Esquema de conexión de tiras led

Para las conexiones de alimentación he decidido no poner los cables al lado tal como se ve en *la Ilustración 5. Esquema de conexión de tiras led*, he distribuido los cables de alimentación en la parte trasera de la base, haciendo unos agujeros en la misma, y con la ayuda de unos pequeños trozos de cable sin recubrimiento hechos en forma de letra T he soldado las líneas de alimentación.

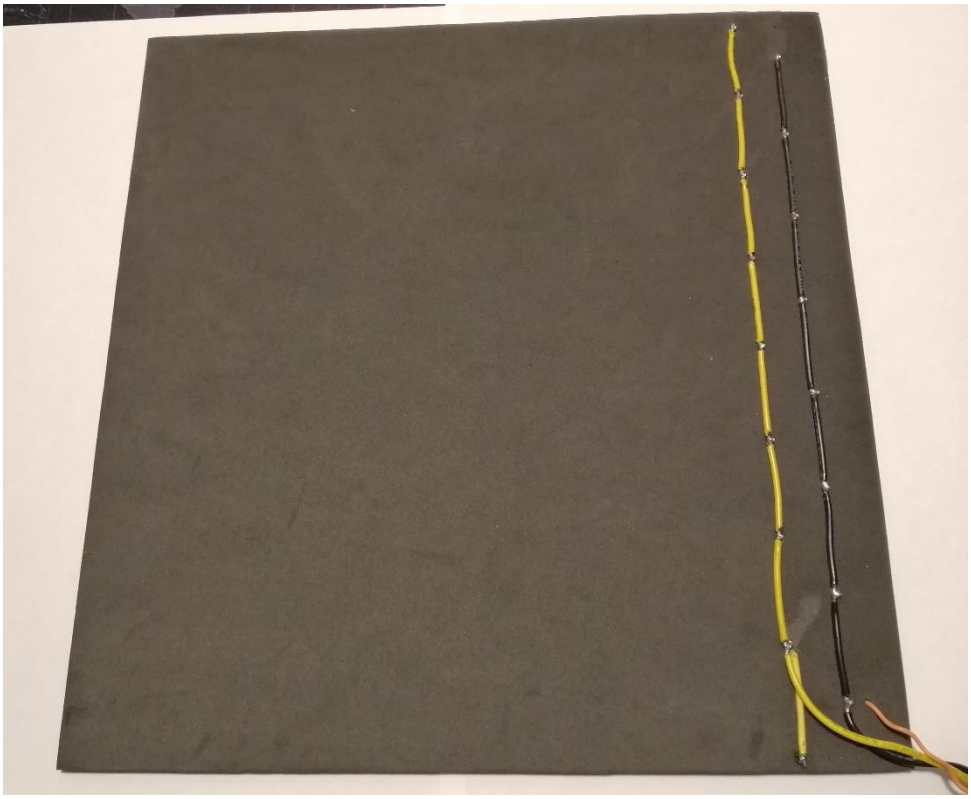


Ilustración 6. Líneas de alimentación soldadas en la parte trasera

Una vez soldado todo paso a fabricar el marco que cubrirá las tiras led, la fabricación del marco consiste en que a partir de la segunda pieza de gomaespuma coartada a principio hay que hacer una marco con cuadrados de tamaño 1cm x 1cm para ello he diseñado una especie de mascara(ver archivo adjunto “Mascara marco”) que se imprime en papel y se pega a la pieza , luego con un bisturí se realizan los recortes de los cuadrados obteniendo el siguiente resultado:

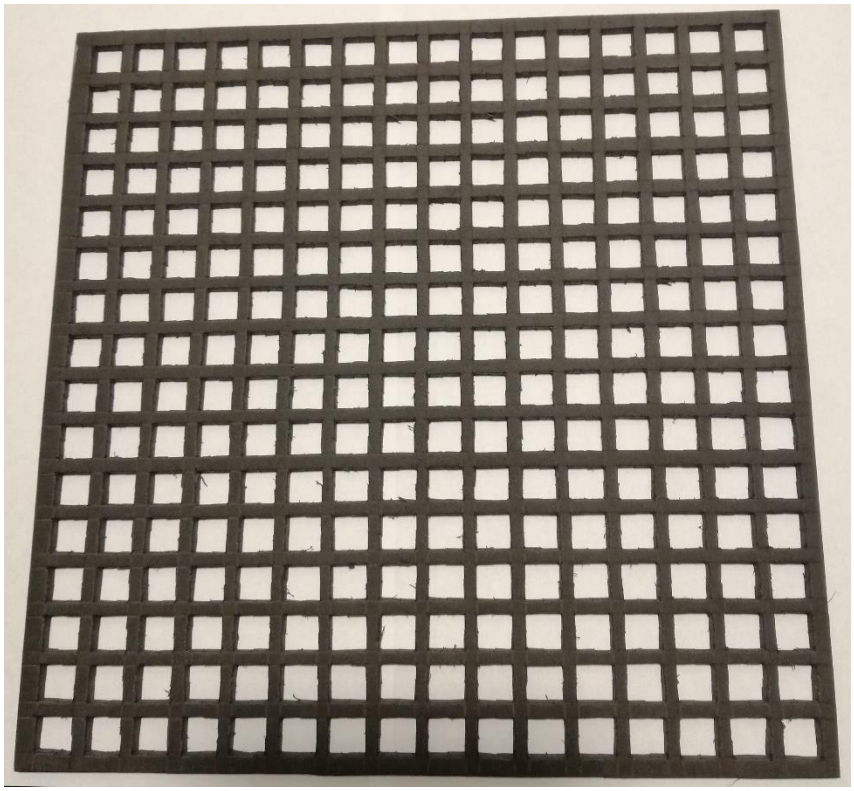


Ilustración 7. Marco frontal

Utilizando pegamento se pega el marco a la base con los leds:

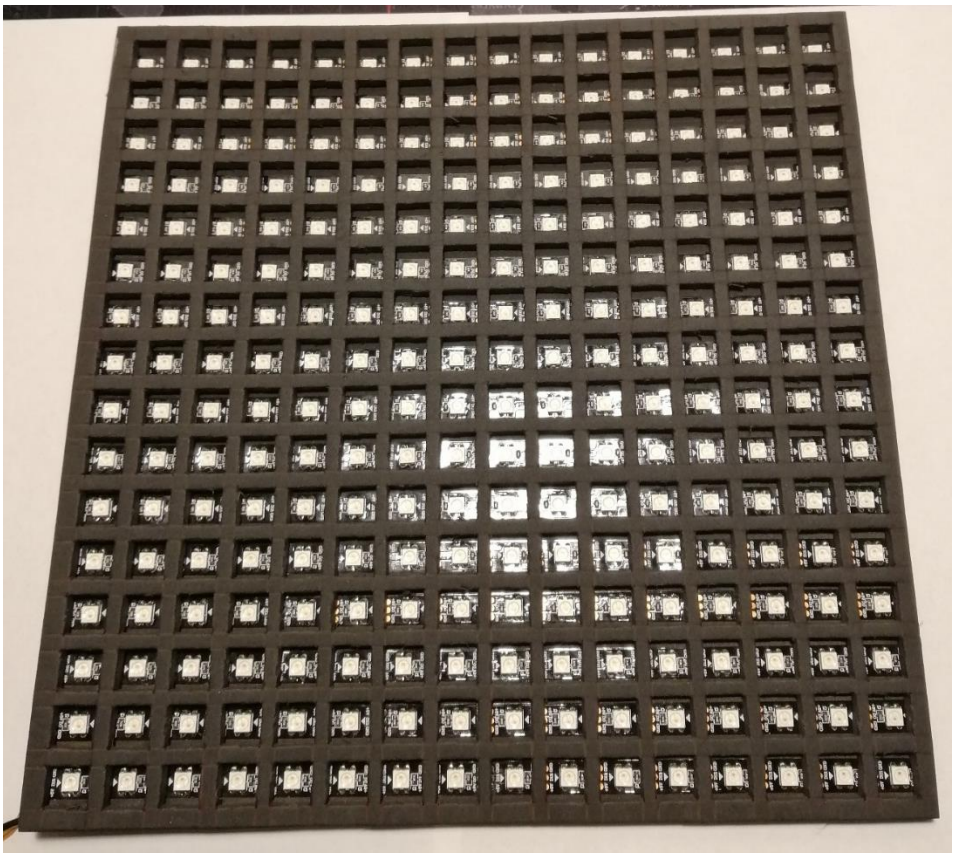


Ilustración 8. Matriz con marco pegado

Tras pegar el marco usando cinta adhesiva de doble cara se pegan por encima del mismo las cubiertas de encuadernación previamente recortadas al tamaño del marco, estas cubiertas ayudan a difuminar la luz y se consigue un efecto visual mejor, aparte de que es protección extra para la estructura.

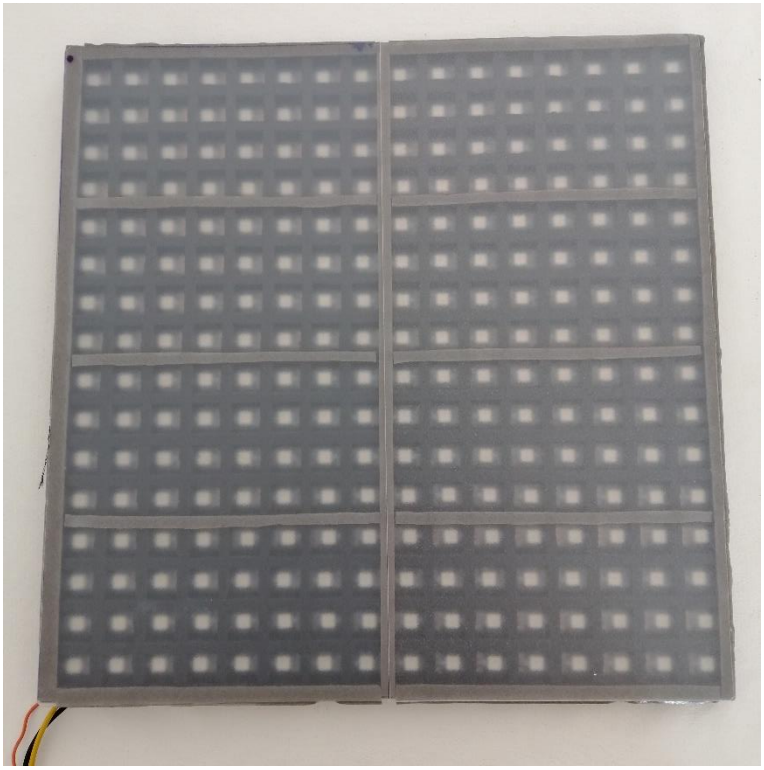


Ilustración 9. Matriz con cubiertas pegadas

Como ultimo paso en la fabricación de la matriz, he hecho una funda de tela para cubrir toda la estructura, la tela en este caso cumple unas funciones estéticas, difumina aun mas la luz y también se usará a la hora de coser la matriz a la mochila.

Ilustración 10. Funda de tela a punto de ser acabada

Una vez acabada la funda y puesta dentro la matriz todo esto se cose a mano a la mochila.

Ilustración 11. Mochila con la matriz led

Problemas que han surgido en esta fase: soldadura de las líneas de alimentación, alineación de leds, corte del marco, problemas con la intensidad de la corriente(insuficiente).

Creación de firmware

Para controlar la matriz led he utilizado la una placa con controlador ESP32

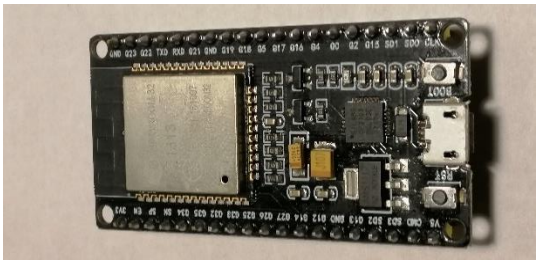


Ilustración 12. Placa controladora ESP32Dev Board

Todo el desarrollo del código se ha realizado en la herramienta Arduino Ide, para trabajar con esta placa hay que importarla primero al entorno de desarrollo, para hacer eso hay que seguir el manual que viene en el repositorio GitHub del fabricante , ver manual.

Para trabajar con la matriz led he utilizado la librería FastLED que proporciona los métodos y funciones básicas para la comunicación y control de la matriz led, al igual que antes, es necesario importar la librería al entrono de desarrollo Arduino.

Tareas que tiene que realizar el firmware:

- Creación de punto de acceso Bluetooth
- Recepción y tratamiento de comandos desde la aplicación móvil
- Creación de utilidades y métodos para crear diferentes efectos visuales
- Visualizar texto recibido desde la aplicación móvil en la matriz led

Para el desarrollo del firmware he estructurado el firmware en diferentes archivos del proyecto para que sea más cómodo trabajar con el código y que sea más entendible. En definitiva, tengo el archivo llamado ***Firmware.ino***, dicho archivo es el principal es donde están definidas la gran mayoría de variables y parámetros para el funcionamiento del dispositivo, los demás archivos contienen diferentes métodos y funciones como por ej: parser, controles de los efectos visuales, corrección de color, funciones para la visualización del texto y efectos visuales.

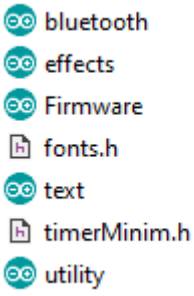


Ilustración 13. Estructura de los archivos del firmware

Resumen descriptivo de los archivos:

Archivo	Descripción
Firmware	Archivo principal, configuraciones y variables generales
bluetooth	Contiene el parser que recibe los parámetros de la app y los gestiona
effects	Contiene todos los métodos de efectos visuales
text	Contiene toda la funcionalidad implementada a para visualizar el texto
utility	Contiene la funcionalidad de corrección de gama, cambio de colores, etc, uso interno
fonts.h	Contiene la codificación de los vocabularios cirilico y latín(ingles) para visualizar el texto
timerMinim.h	Contiene la clase timer que sirve para controlar los tiempos(velocidad) de visualización de los efectos.

Explicación del código principal:

A principio del código se definen todas las librerías que se usaran en el programa

```
1  #include <FastLED.h>
2  #include "BluetoothSerial.h"
3  #include "fonts.h"
4  #include "timerMinim.h"
5
```

Luego defino todas las variables y parámetros que son vitales para el funcionamiento del programa, por ej el ancho y alto de la matriz, orden de colores, y el pin de datos de la placa

```
6  //matriz
7  #define WIDTH 16 // ancho de la matriz
8  #define HEIGHT 16 // alto
9
10 //leds
11 #define NUM_LEDS WIDTH * HEIGHT // numero de leds total en la matriz
12 #define DATA_PIN 12  // pin de datos para la comunicacion con los leds
13 #define COLOR_ORDER GRB
14 #define CURRENT_LIMIT 2000 // limite de corriente en miliamperios, autoajuste de brillo
15
16 //array de leds
17 CRGB leds[NUM_LEDS];
```

A continuación, vienen definidos los parámetros y variables para los efectos visuales y funcionamiento correcto del programa

```
19 //parametros para efectos
20 #define EFFECT_SPEED 100  // velocidad estandar para efectoss
21 #define BRIGHTNESS 100    // brillo estandar, valores entre 0 y 255
22 #define TEXT_SPEED 100    // velocidad de texto , en ms
23
24 uint32_t globalColor = 0x00ff00;
25 int currentMode = 255; //modo de funcionamiento 0 - demo, 1 - efectos, 2 - texto
26 int currentBrightness = BRIGHTNESS;
27 int currentSpeed = EFFECT_SPEED;
28 int currentEffect = 255;
29 int extraParameter = 0;
30 boolean effectUpdate = false;
31 boolean effectStart = false;
32 boolean effectIsRunning = false;
33 boolean loadingFlag = true;
34 boolean extraFlag = false;
35 boolean colorFlag = false;
36 String text = "";
37
```

También defino los objetos timer y bluetooth

```
38  //objeto timer
39  timerMinim effectTimer(currentSpeed);
40  timerMinim scrollTimer(currentSpeed);
41
42
43  //objeto bluetooth
44  BluetoothSerial SerialBT;
45
```

Para controlar cuando se conecta un usuario con la aplicación del teléfono he usado uno de los eventos internos de la controladora ESP32.

```
46  void callback(esp_spp_cb_event_t event, esp_spp_cb_param_t *param) {
47    if (event == ESP_SPP_SRV_OPEN_EVT) {
48      Serial.println("Client Connected");
49    } else if (event == ESP_SPP_CLOSE_EVT) {
50      Serial.println("Client disconnected");
51    }
52  }
```

El resultado que genera este evento es simplemente informativo y se puede ver en la consola serie, es para tener conocimiento de cuando un usuario se conecta o se desconecta.

A continuación, vienen las partes estándar de cualquier sketch *setup ()* y *loop ()*, en la parte de setup se define el tipo de controladores que se usan en la matriz led, se inician los objetos de las clases bluetooth y timer, se configuran algunos parámetros para la matriz led, y se pone en funcionamiento el sistema de limitación de potencia en función de la intensidad que consume la matriz, si la intensidad se pasa del limite establecido se autoajusta el brillo y así la matriz sigue funcionando y no se apaga.

```
54  void setup() {
55    Serial.begin(115200);
56    //bluetooth setup
57    SerialBT.register_callback(callback);
58    if (!SerialBT.begin("8BitBag")) {
59      Serial.println("Error bluetooth initializing");
60    } else {
61      Serial.println("All fine, bluetooth initialized");
62    }
63
64    //leds setup
65    FastLED.addLeds<NEOPIXEL, DATA_PIN>(leds, NUM_LEDS).setCorrection( TypicalLEDStrip );
66    FastLED.setBrightness(BRIGHTNESS);
67    if (CURRENT_LIMIT > 0) {
68      FastLED.setMaxPowerInVoltsAndMilliamps(5, CURRENT_LIMIT);
69    }
70    FastLED.clear();
71    FastLED.show();
72
73    randomSeed(analogRead(0) + analogRead(1)); // generar numeros aleatorios
74  }
```

Ilustración 14. parte de setup

```
76  void loop() {
77    parser();
78    if (effectStart) {
79      if (currentMode == 1) {
80        if (effectTimer.isReady()) {
81          effectsRoutine();
82        }
83      }
84      if (currentMode == 2) {
85        textRoutine(text, globalColor);
86      }
87    } else {
88      clearRoutine();
89      effectIsRunning = false;
90    }
91  }
```

Ilustración 15. Parte de loop

Todo el funcionamiento del programa se basa en que en función de los parámetros que reciba el parser vía bluetooth se van llamando diferentes rutinas para efectos y se van estableciendo diferentes modos de funcionamiento, por ej. el modo 2(texto) no sigue la misma lógica de funcionamiento que el modo 1(efectos visuales).

Problemas encontrados en el proceso de desarrollo: recepción de una cadena de caracteres, colores erróneos.

Creación de aplicación para teléfono móvil.

Para controlar el dispositivo de manera remota y cómoda he decidido crear una aplicación para la plataforma Android, para esto usare el entorno de desarrollo Android Studio, programándola en lenguaje java.

Objetivos que tendrá que cumplir la aplicación:

- Detectar y reconocer dispositivos visibles por protocolo bluetooth alrededor
- Establecer conexión con el dispositivo seleccionado de la lista
- Realizar comunicación con el dispositivo (enviar comandos)

Para conseguir la funcionalidad básica de la aplicación y cumplir con los dos primeros objetivos marcados he programado una aplicación base que simplemente realizaba un escaneo, creación de lista de dispositivos y conexión al dispositivo seleccionado. Esta parte de programación fue la que ocupo aproximadamente 40% del tiempo total que he empleado en el proyecto, cuando logre cumplir los dos primeros objetivos implemente toda la lógica para poder enviar parámetros y comandos a la placa controladora y como último paso implemente una interfaz gráfica usable, no muy compleja para poder usar cómodamente la aplicación.

A continuación, detallo algunas capturas de pantalla de la aplicación

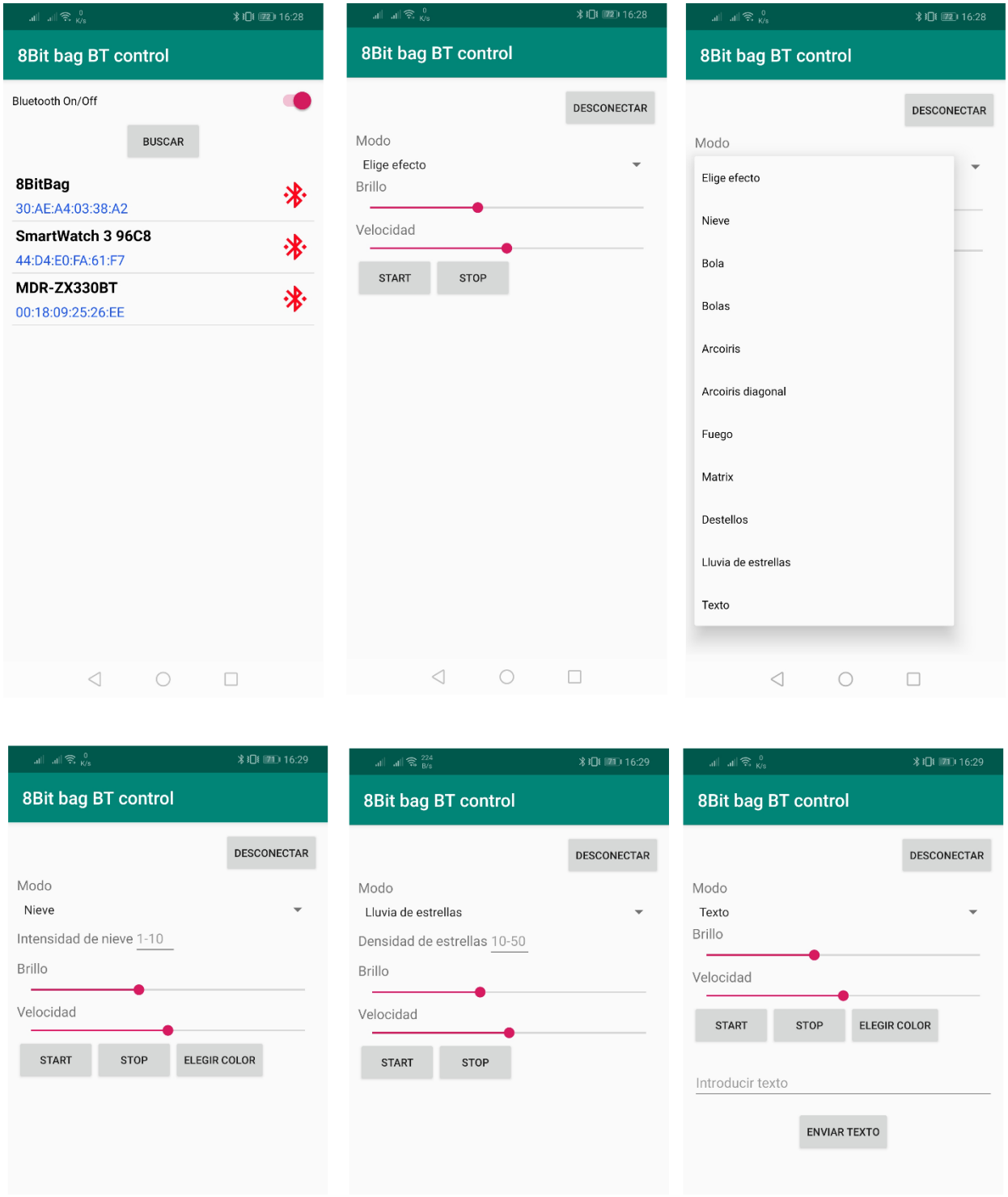


Ilustración 16. Capturas de pantallas de la aplicación

Como podemos observar en Ilustración 16. Capturas de pantallas de la aplicación , algunos de los efectos aparte de los controles estándar de “Brillo” y “Velocidad” tienen opciones extra que permiten cambiar parámetros en esos efectos en concreto.

Problemas encontrados en esta fase: no se puede crear un cuadro de selección de color con herramientas nativas de Android studio.

Solución de problemas encontrados

Inestabilidad de funcionamiento de la placa: dado que la matriz led consume mucha potencia y produce saltos bastante importantes de tensión la placa controladora funcionaba de manera muy inestable, esto se ha solucionado incorporando un condensador electrolitico de 6.3 v y 470mF en la línea de alimentación de la placa, con esto se consigue dar estabilidad a la placa y reducir los saltos de tensión.

Soldadura de las líneas de alimentación: para las líneas de alimentación he usado cables de bastante y grosor y rigidez ya que tienen que soportar hasta 3A de intensidad, dado esto no era factible realizar soldaduras laterales como las que se ven en Ilustración 5. Esquema de conexión de tiras led, ya que ocuparían un espacio lateral considerable y visualmente quedarían mal, la solución que he implementado ha sido fabricar una serie contactos en forma de letra T, soldarlos a las líneas de alimentación y pasarlos por unos agujeros en la base de gomaespuma, posteriormente en la parte trasera de la base he realizado las soldaduras con los cables , ver Ilustración 6. Líneas de alimentación soldadas en la parte trasera .

corte del marco: inicialmente fabriqué una herramienta a partir de una espátula en forma de cuadrado 1cm x 1 cm, con la idea de hacer los cortes con esa herramienta, pero al ver que no cortaba bien y estropeaba el material decidí hacer los corte a mano con un bisturí, en total 1024 cortes realizados en 2h.

problemas con la intensidad de la corriente(insuficiente): en principio me encontré el problema de que la matriz led no funciona adecuadamente si se le poner mas de la mitad del brillo, ese problema se debe a que dichas tiras led están pensadas para trabajar con fuentes de alimentación que puedan proporcionar al menos 3A de intensidad de corriente, dado que mi proyecto usa un PowerBank como fuente de alimentación me he visto obligado a reducir mediante software el consumo energético a máximo de 2A que es un valor más común para los PowerBank.

recepción de una cadena de caracteres: a la hora de implementar el parser para recibir e interpretar los comandos enviados desde la app del teléfono me tope con el problema de recepción de una cadena de caracteres y de que el programa no detectaba donde acababa dicha cadena de caracteres, la solución fue definir unos caracteres especiales para inicio y final de cada comando, de este modo he conseguido tener delimitado el inicio y final de cada comando y poder tratarlo de manera adecuada.

colores erróneos: a la hora de enviar los códigos de colores desde la aplicación móvil la matriz led interpretaba algunos de esos colores de manera errónea , para solventar ese problema he implementado una función llamada *gammaCorrection* que se encarga de corregir ese error, lo que hace es descomponer el código de color en bytes hexadecimales y los compara con arrays de valores de color para cada byte, Red, Green, Blue, después de obtener nuevos valores de esos bytes , vuelve a componer el código de color ya corregido.

Mejoras

El dispositivo o el sistema implementado tiene mucho margen de mejora, y esta pensado así a propósito, ya que cualquier persona puede usarlo como base en su propio proyecto, crear nuevos efectos visuales, añadir más módulos a la placa, como por ej. giroscopios, sensores de movimiento, sensores de luz, y un largo etc.

Personalmente a mí me gustaría mejorar los siguientes aspectos del sistema:

Implementar una comunicación bidireccional dispositivo <-> teléfono móvil.
Aprovechando que la placa tiene WIFI implementar un sistema de actualizaciones vía OTA.
Reducir el consumo energético del dispositivo.
Hacer que la matriz sea modular.
Pixelizar imágenes en la matriz.
Dibujar en la matriz.

Costes

Los costes en materiales para fabricar este dispositivo han sido los siguientes:

Pieza	Valor
Tira de diodos LED RGB 5V, 5m 60leds/m IP30	13.81 €
ESP32 Dev Board	4.78 €
Plancha de gomaespuma 40cm x 60cm x 5mm	2 €
Condensador electrolítico 6.3v 470mF	0.60 €
Conector USB macho tipo A	0.20 €
Cubierta encuadernación DIN A4 0.45mm plástico - 2 unidades	0.80 € / unidad
Cinta adhesiva doble cara 3M 3metros – 2 rollos	0.90 € / unidad
Cable – 1.5 metros	1 €
Resistencia 200 – 400 Ω	0.20 €
Mochila – 1 unidad	15 €
Total	40.99 € IVA incl

Conclusiones

Trabajar en este proyecto me ha hecho aprender mas cosas sobre los dispositivos empotrados, aprender a crear aplicaciones para teléfonos móviles Android, aprender infinidad sobre los efectos luminosos, como se comportan y come se pueden crear. En realidad, este sistema se puede interpretar como un lienzo en blanco ya que cualquiera con conocimientos de programación puede utilizarlo y mejorarlo, crear sus propios efectos visuales o incorporarlo a sus propios sistemas como un añadido.

Objetivos cumplidos

Considero todos los objetivos marcados al principio de esta memoria como cumplidos ya que se ha obtenido un producto final funcional y adecuado a los objetivos.