# SPRINT 8. ESP32

## Configuración inicial

Para comenzar he intentado subir un simple código al ESP para comprobar que funcione correctamente. Me he encontrado con multitud de problemas:

- He optado en un principio a conectarlo por el USB-C de la placa a mi ordenador pero el ordenador no detectaba la placa. Accediendo a "Administrador de Dispositivos" y a los puertos COM, no habia nada detectado. Para solucionarlo, he buscado un cable micro usb y ya se detectaba algo pero mal.
- Una vez el ordenador detectara algo, me he acordado que existía un software para placas que no eran propias de Arduino. Este driver se llama CH341SER. Una vez instalado el driver, ya se reconocía la placa correctamente.
- Cuando fui a subir el simple script a la placa desde el software de Arduino, cuando he ido a seleccionar la placa me salía "Unknown" pero si que reconociendo que había algo en el COM3. Para solucionar esto, he tenido que acceder a File > Preferences > Additional Boards Manager URLs y poner esta URL https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json Por lo que he estado leyendo es necesaria para que el software de Arduino pueda tener la opción de descargar los drivers de la placa.
- Tras esto, accedí a Board Manager para instalar los drivers del ESP32. Estos se llamaban esp32 by Espressif Systems. (Tampoco los pude instalar a la primera. Tras intentarlo múltiples veces e incluso cambiar la versión de estos drivers a una anterior pero sin éxito, se me ocurrió reiniciar el ordenador y así conseguí instalarlos).
- Despues he seleccionado en Board > esp32 > ESP32S2 Dev Module (Seleccione esa sin conocimiento de causa, simplemente era el nombre que más se parecía) y por fín he conseguido terminar el setup.

## El código

Ahora toca la parte del código siguiendo las especificaciones que se han hablado previamente. He de decir que bajo mis mínimos conocimientos sobre C++, la sintaxis del código ha sido hecha por Gemini y yo he ido probando y testeando que todo funcione correctamente. En primer lugar, he tenido que instalar las biblioteca utilizadas (ya mencionadas en el sprint anterior). Esto ha funcionado todo sin ningún problema. Después he ido a probar el código que me había proporcionado Gemini con las especificaciones que yo le había dado pero me ha saltado un error de versiones entre la librería ESPAsyncWebServer y la placa. Por lo visto, la librería está llamando a funciones (_ret) que la placa aún no conoce. La solución que he hecho ha sido ir al código de la propia librería y modificarlo, quitandole esas funciones. Básicamente, ha quedado así:

```
// mbedtls_md5_starts_ret(&_ctx);
// mbedtls_md5_update_ret(&_ctx, data, len);
// mbedtls_md5_finish_ret(&_ctx, _buf);
mbedtls_md5_starts(&_ctx);
mbedtls_md5_update(&_ctx, data, len);
mbedtls_md5_finish(&_ctx, _buf);
```

Y contra todo pronóstico, ha funcionado y he logrado que la aplicación compile. Ahora he ido a subir el código al ESP y no ha funcionado. Como comente al principio, habia seleccionado la placa ESP32S2 Dev Module sin motivo y me ha arrojado un código en la subida diciendome que esa no era la placa correcta. Justo en debajo de la opción que había seleccionado antes encontré una que se llamaba ESP32 Dev Module. Selecciono esa, lo subo a la placa y funcionó. Ahora toca meter los archivos que tengo dentro de la carpeta ./data/ que son el html, css, js y json dentro del ESP. Para ello, la idea era meter esos archivos utilizando SPIFSS o LittleFS. He visto que es mas común utilizar LittleFS a sí que usaremos este. En primer lugar, me he tenido que descargar la version .vsix de github: https://github.com/earlephilhower/arduino-littlefs-upload/releases e instalarla en el Arduino. Esto era un lio impresionante que tras varias horas no he conseguido que funcione a sí que de manera provisional he optado por integrar el html, css y js dentro del StockManager.ino. Más problemas que he tenido han sido con la pantalla. Cuando subía algo al ESP, así fuera un simple código de ejemplo, la pantalla solamente se ponía en blanco. Esto es porque la versión de la pantalla es una más moderna de la que utiliza TFT_eSPI, en este caso, con el chip ST7789 en lugar del antiguo ILI9341. Para arreglar este problema, he modificado el archivo User_Setup.h de la librería TFT_eSPI eliminando todo lo que contenía y escribiendo lo siguiente:

```
#define USER_SETUP_INFO "User_Setup"

// --- AQUÍ ESTÁ EL CAMBIO CLAVE ---
#define ST7789_DRIVER      // Usamos el driver ST7789 en lugar del ILI9341
#define TFT_RGB_ORDER TFT_BGR  // Orden de colores BGR (Blue-Green-Red)
#define TFT_INVERSION_ON   // Invierte los colores (necesario en estas pantallas)

// Resolución de la CYD
#define TFT_WIDTH  240
#define TFT_HEIGHT 320

// Configuración de Pines (Esto no cambia)
#define TFT_MISO 12
#define TFT_MOSI 13
#define TFT_SCLK 14
#define TFT_CS   15
#define TFT_DC   2
#define TFT_RST  -1
#define TFT_BL   21  // Luz de fondo

// Fuentes de letra
#define LOAD_GLCD
#define LOAD_FONT2
#define LOAD_FONT4
#define LOAD_FONT6
#define LOAD_FONT7
#define LOAD_FONT8
#define LOAD_GFXFF
#define SMOOTH_FONT

// Velocidad SPI
#define SPI_FREQUENCY  55000000
#define SPI_READ_FREQUENCY  20000000
#define SPI_TOUCH_FREQUENCY  2500000
```

Y ahora si que he conseguido que el código sencillo de la pantalla funcionara correctamente, cambiara de color la pantalla y me mostrara caracteres. A partir de aqui, me ha sido imposible seguir avanzando. Cada vez que intentaba subir algun minimo script a la placa no paraba de conectarse y descontarse, reinicie el ordenador, reinicie la placa, probe otros scripts, deje de utilizar la libreria modificada, aumente la memoria de la placa en las opciones del software y de ninguna manera he conseguido mostrar de nuevo algo en la placa.