

Comunicació bidireccional entre ordinador i ESP32 mitjançant port sèrie USB

Ruben Santamaria i Juan Vidal

2025-2026

SMX2B

Dani Soldevila

INDEX:

1. Títol i objectiu de la pràctica.....	3
2. Descripció general de la pràctica.....	4
3. Conceptes clau.....	4
3.1 Què és una classe?.....	4
3.2 Què és un objecte?.....	4
3.3 Què és un mètode?.....	5
3.4 Relació amb l'objecte Serial.....	5
4. Esquema de connexions (Fritzing).....	5
5. Diagrama de flux del programa.....	6
6. Flux de dades i procés de lectura/escriptura.....	7
Diagrama visual del flux de dades.....	7
7. Explicació tècnica de les funcions utilitzades.....	7
Lectura de dades.....	8
Serial.available().....	8
Serial.readStringUntil('\n').....	8
Processament de dades.....	8
trim().....	8
equalsIgnoreCase().....	8
Enviament de respostes.....	8
println().....	8
8. Proves realitzades.....	9
IMATGES GITHUB.....	10

1. Títol i objectiu de la pràctica

L'objectiu principal d'aquesta pràctica ha estat aprendre a establir comunicació bidireccional entre un ordinador i una placa ESP32 utilitzant el port sèrie USB.

Hem treballat l'enviament i recepció de dades en format text, i hem implementat un sistema capaç d'interpretar ordres escrites des del Monitor Sèrie per controlar l'estat d'un LED.

2. Descripció general de la pràctica

En aquesta pràctica hem connectat un LED a un pin digital de l'ESP32.

Des de l'ordinador, mitjançant el Monitor Sèrie, enviem ordres textuais:

- **ON → encendre el LED**
- **OFF → apagar el LED**

El microcontrolador rep la informació, la processa i executa una acció en funció del text rebut. A més, envia una resposta de confirmació al Monitor Sèrie.

Això ens ha permès entendre el funcionament real de la comunicació bidireccional.

3. Conceptes clau

3.1 Què és una classe?

Una classe és una plantilla que defineix característiques i comportaments. És com un model que permet crear objectes amb propietats i funcions determinades.

3.2 Què és un objecte?

Un objecte és una instància d'una classe. És l'element real que utilitzem en el programa.

3.3 Què és un mètode?

Un mètode és una funció que pertany a una classe i que defineix una acció que pot realitzar un objecte.

3.4 Relació amb l'objecte Serial

En el nostre programa utilitzem l'objecte Serial.

- **Serial és un objecte de la classe `HardwareSerial`.**
- **Té mètodes com:**
 - **`begin()`**
 - **`available()`**
 - **`readStringUntil()`**
 - **`println()`**

Aquests mètodes ens permeten enviar i rebre dades a través del port sèrie.

4. Esquema de connexions (Fritzing)

Connexions realitzades:

- Pin 25 de l'ESP32 → Resistència (220Ω) → Ànode del LED
- Càtode del LED → GND

L'esquema en Fritzing mostra:

ESP32

|
Pin 25 — Resistència — LED — GND

Aquest esquema garanteix que el LED funcioni correctament i que no es danyi per excés de corrent.

5. Diagrama de flux del programa

Per fer el diagrama de flux hem seguit aquests passos:

1. Identificar inici del programa (setup).
2. Configuració inicial.
3. Bucle infinit (loop).
4. Comprovació de dades disponibles.
5. Lectura de dades.
6. Comparació d'ordres.
7. Execució d'acció.

8. Retorn al loop.

Flux simplificat:

Inici



Configuració Serial i LED



Comprovar Serial.available()



Hi ha dades?

→ **No** → **Tornar al loop**

→ **Sí** → **Llegir text**



Comparar amb ON/OFF



Encendre / Apagar / Error



Tornar al loop

6. Flux de dades i procés de lectura/escriptura

Diagrama visual del flux de dades

Monitor Sèrie (PC)



USB



ESP32



Pin digital



LED

Resposta:

ESP32



USB



Monitor Sèrie

Això demostra que la comunicació és bidireccional.

7. Explicació tècnica de les funcions utilitzades

Lectura de dades

Serial.available()

Comprova si hi ha dades pendents de lectura al buffer.

Si retorna un valor major que 0, significa que hi ha informació disponible.

Serial.readStringUntil('\n')

Llegeix el text rebut fins detectar un salt de línia (quan es prem Enter).

Processament de dades

trim()

Elimina espais en blanc i salts de línia al principi i al final del text.

equalsIgnoreCase()

Permet comparar dues cadenes ignorant majúscules i minúscules.

Exemple:

`ordre.equalsIgnoreCase("ON")`

Això evita errors si l'usuari escriu "on", "On" o "oN".

Enviament de respostes

`println()`

Envia un missatge pel port sèrie i afegeix un salt de línia automàtic.

Això ens permet veure clarament les respostes al Monitor Sèrie.

8. Proves realitzades

Hem realitzat diferents proves:

Ordre enviada	Resultat
ON	LED encès
OFF	LED apagat

on	LED encès (gràcies a equalsIgnoreCase)
qualsevol text	Missatge d'error

Les proves han confirmat que:

- **La comunicació funciona correctament.**
- **El sistema interpreta les ordres.**
- **El LED respon immediatament.**
- **Es mostra confirmació al Monitor Sèrie.**

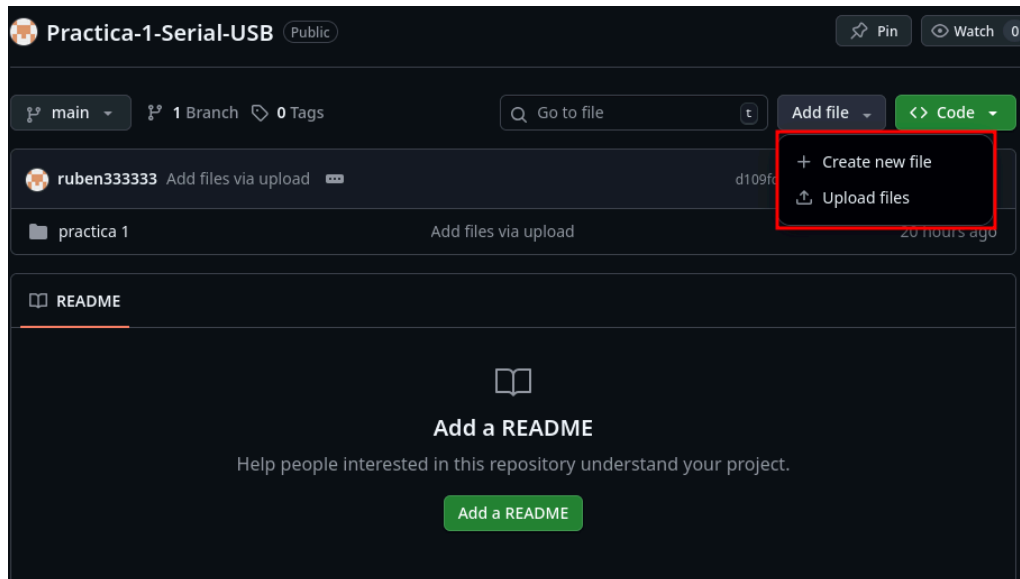
9. Vídeo demostratiu del funcionament

T'hem penjat el video al GitHub.

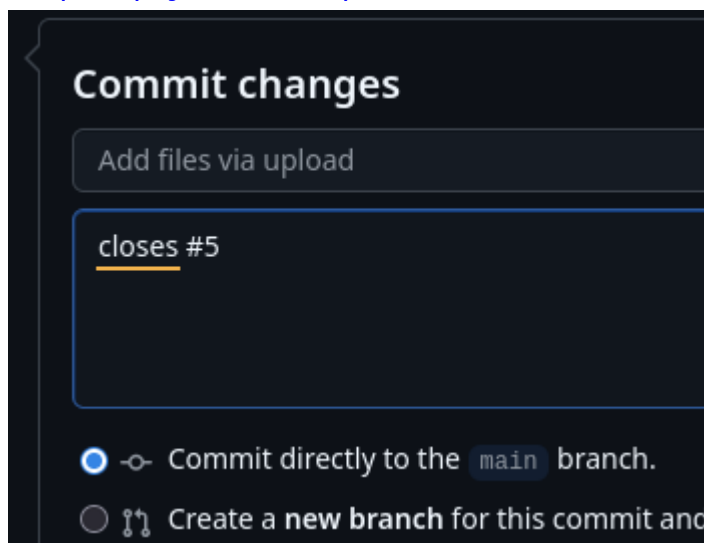
[GitHub](#)

IMATGES GITHUB

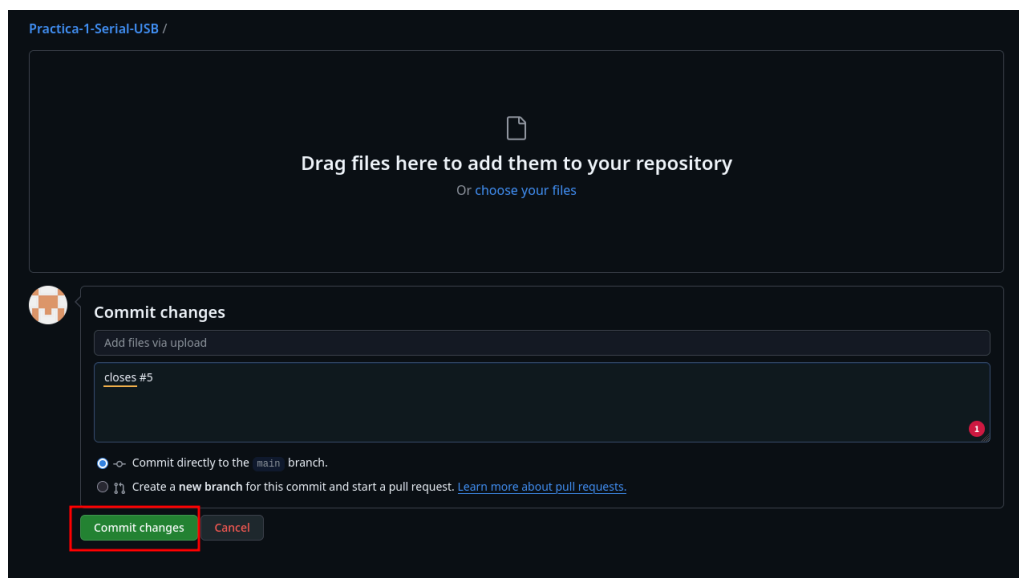
Per pujar un arxiu a un issue es fa de la següent manera



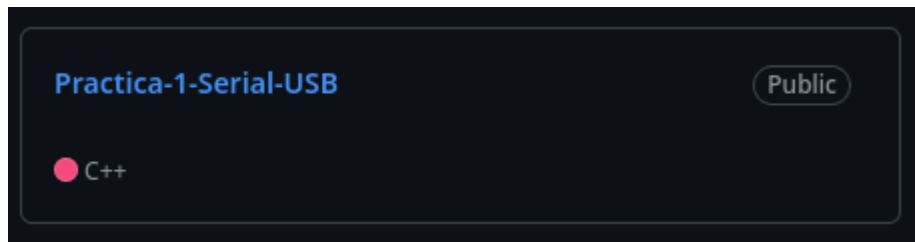
Després pujes l'arxiu i poses closes # i el número del issue en aquest cas es 5



I després el guardem

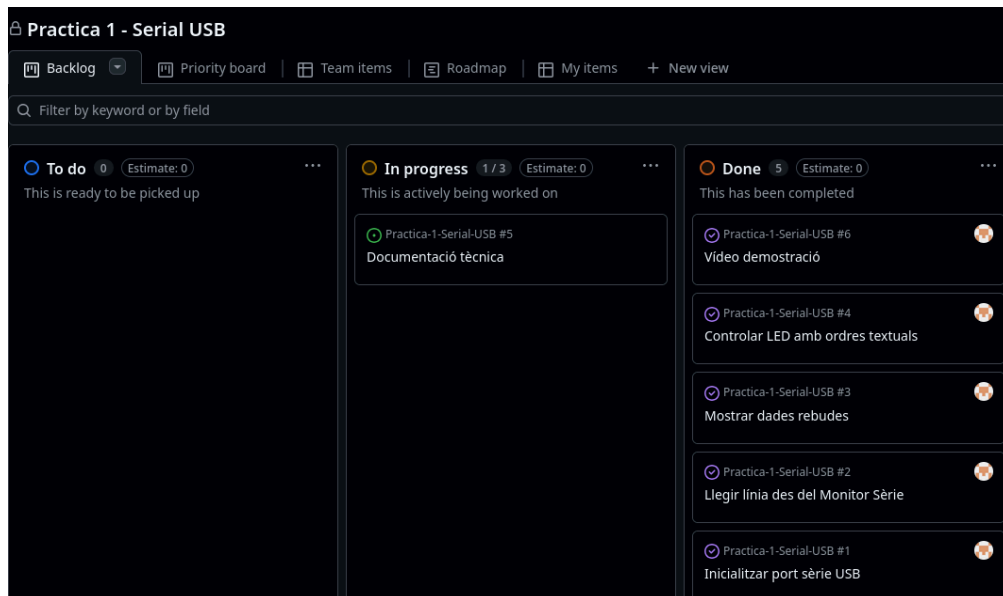


Aquest és el projecte

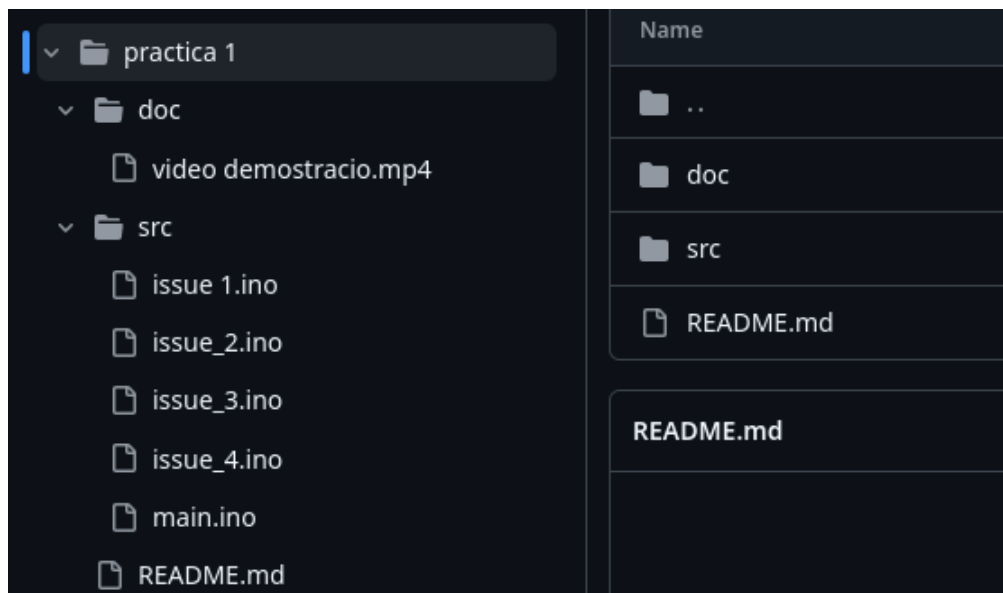


Dins del projecte tenim:


- Issues






- Tenim també la ruta de carpetes on està tot guardat




- Tenim cada issue amb el seu codi


Practica-1-Serial-USB / practica 1 / src / issue 1.ino 

 ruben333333 Add files via upload 

Code Blame 11 lines (8 loc) · 151 Bytes 

```
1  const int LED_PIN = 25;
2
3  void setup() {
4      Serial.begin(115200);
5      delay(500);
6      Serial.println("Sistema llest. Escribe ON o OFF:");
7  }
8
9  void loop() {
10
11  }
```


Practica-1-Serial-USB / practica 1 / src / issue_2.ino 

 ruben333333 Add files via upload 

Code Blame 12 lines (11 loc) · 228 Bytes 

```
1  void setup() {
2      Serial.begin(115200);
3      delay(500);
4      Serial.println("Sistema llest. Escribe ON o OFF:");
5  }
6
7  void loop() {
8      if (Serial.available() > 0) {
9          String ordre = Serial.readStringUntil('\n');
10         ordre.trim();
11     }
12 }
```


Practica-1-Serial-USB / practica 1 / src / issue_3.ino

 ruben333333 Add files via upload

Code Blame 18 lines (16 loc) · 375 Bytes

```
1  const int LED_PIN = 25;
2
3  void setup() {
4      Serial.begin(115200);
5      delay(500);
6      pinMode(LED_PIN, OUTPUT);
7      digitalWrite(LED_PIN, LOW);
8      Serial.println("Sistema llest. Escriu ON o OFF:");
9  }
10
11 void loop() {
12     if (Serial.available() > 0) {
13         String ordre = Serial.readStringUntil('\n');
14         ordre.trim();
15         ordre.toUpperCase();
16         Serial.println("Rebut: " + ordre);
17     }
18 }
```

Practica-1-Serial-USB / practica 1 / src / issue_4.ino

 ruben333333 Add files via upload

Code Blame 30 lines (27 loc) · 645 Bytes

```
1  const int LED_PIN = 25;
2
3  void setup() {
4      Serial.begin(115200);
5      delay(500);
6      pinMode(LED_PIN, OUTPUT);
7      digitalWrite(LED_PIN, LOW);
8      Serial.println("Sistema llest. Escriu ON o OFF:");
9  }
10
11 void loop() {
12     if (Serial.available() > 0) {
13         String ordre = Serial.readStringUntil('\n');
14         ordre.trim();
15         ordre.toUpperCase();
16         Serial.println("Rebut: " + ordre);
17
18         if (ordre == "ON") {
19             digitalWrite(LED_PIN, HIGH);
20             Serial.println("LED ENCÈS");
21         }
22         else if (ordre == "OFF") {
23             digitalWrite(LED_PIN, LOW);
24             Serial.println("LED APAGAT");
25         }
26         else {
27             Serial.println("Ordre desconeguda");
28         }
29     }
30 }
```

- També tenim un issue amb el vídeo de demostració

