Universidad del Valle de Guatemala

Departamento de Electrónica

Electrónica Digital 2

Pablo Mazariegos

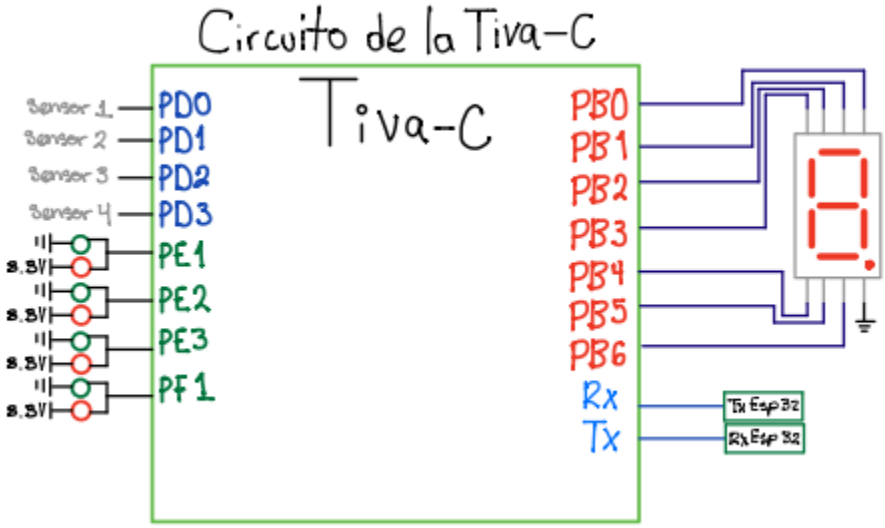
Proyecto # 4

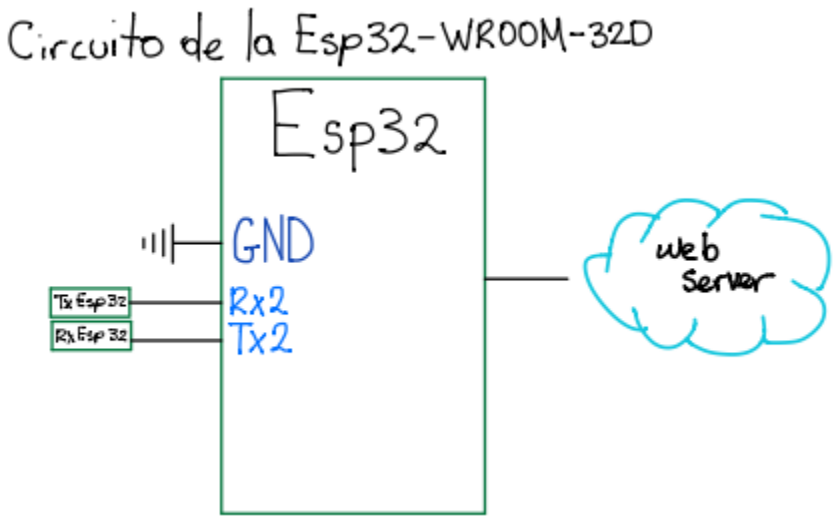
Parqueo con Web server

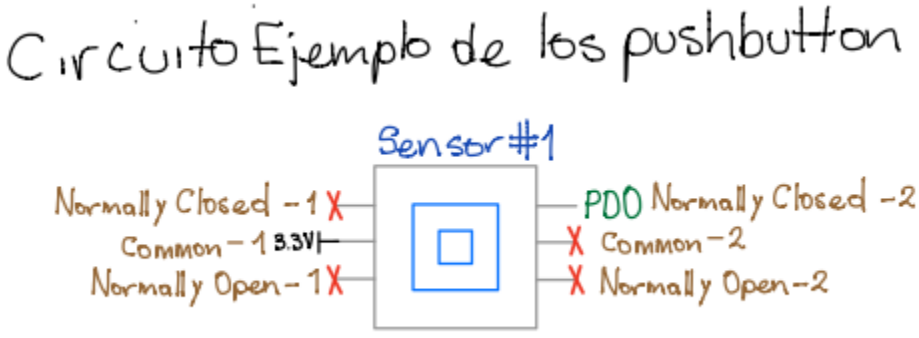
Mariandrée Rivera 18178

25 de mayo de 2021, Ciudad de Guatemala, Guatemala.

Circuitos Utilizados:





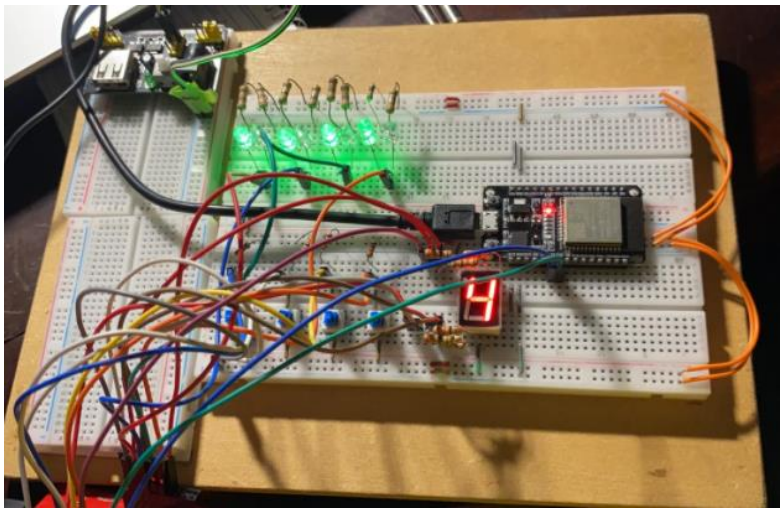


Datos:

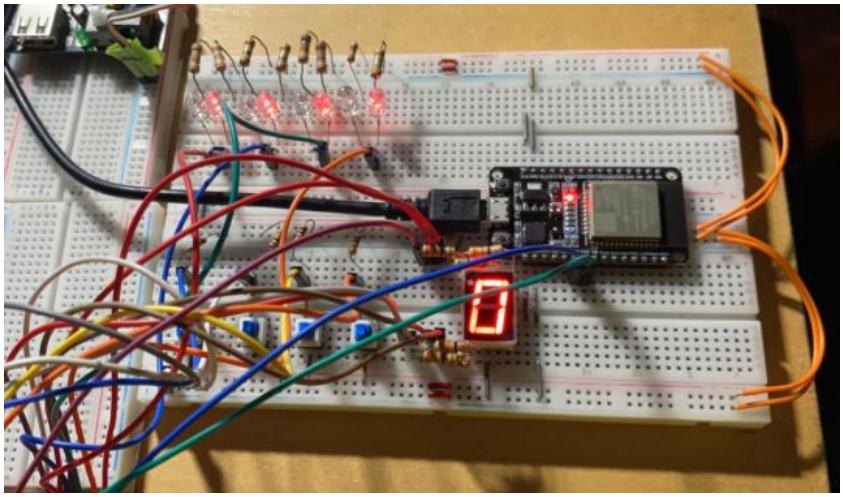
- S#, almacena el dato que recibe desde el pushbutton.
- Lugares, almacena el dato de la cantidad de parqueos disponibles.
- ST, desplaza las variables para que se pueda realizar la comunicación UART.
- numero, contiene el valor que se tiene que desplegar en el 7 segmentos.
- n_p, numero de parqueos disponibles en el web server.

Gráficos:

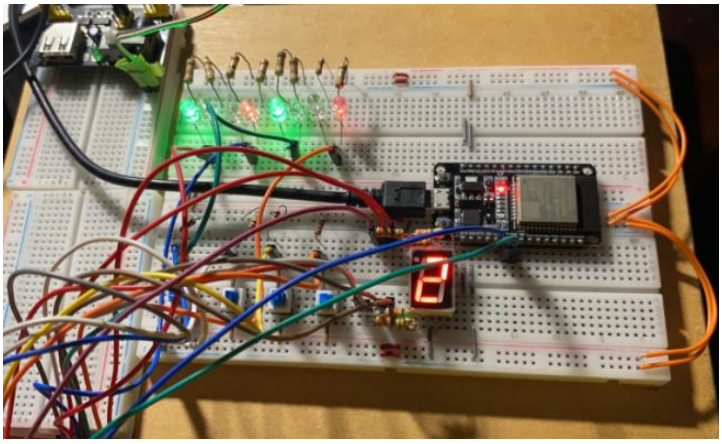
Implementacion fisica todos los parqueos libres:



Implementacion fisica todos los parqueos ocupados:



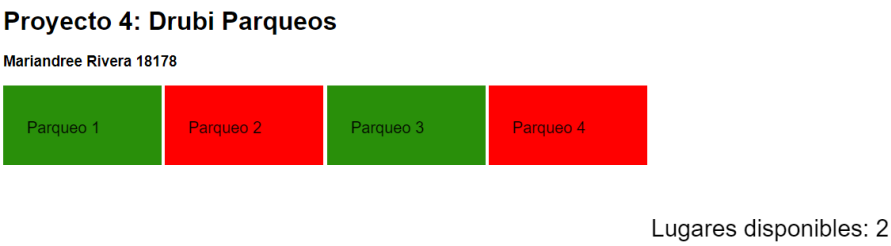
Implementacion fisica a la mitad de la capacidad:



Pagina web todos los parqueos libres:

Pagina web todos los parqueos ocupados:

Pagina web con disponibilidad a la mitad:



Explicación:

Este proyecto consistía en que debíamos elaborar un sistema de parqueo como se realiza en muchos centros comerciales actuales, mediante sensores o indicadores que se encargaran de enviar una señal cuando un carro llegue a los distintos lugares. Cuando el carro llega a uno de los parqueos disponibles la led que está asociada a ella debe de cambiar de verde a roja, de manera que de forma local se pudiera visualizar de que esta ocupado. También se tiene un 7 segmentos que se encargaba de mostrar la cantidad de lugares disponibles.

En adicional también por medio de un modulo de wifi, se debía de crear una pagina de internet que nos desplegara la misma información, los lugares disponibles de alguna forma grafica y la cantidad de parqueos disponibles también.

Códigos:

Código la TIVA-C:

```
//*****************************************************************************
//  Alumno: Mariandree Rivera
//  Carnet: 18178
//  Proyecto # 4, Parqueo
//*****************************************************************************


//*****************************************************************************
// Librerias
//*****************************************************************************
#include <stdint.h>
#include <stdbool.h>
#include "inc/tm4c123gh6pm.h"
#include "inc/hw_memmap.h"
#include "inc/hw_types.h"
#include "inc/hw_ints.h"
#include "inc/hw_gpio.h"
#include "driverlib/sysctl.h"
#include "driverlib/interrupt.h"
#include "driverlib/gpio.h"
#include "driverlib/timer.h"
#include "driverlib/uart.h"
#include "driverlib/pin_map.h"

#define XTAL 16000000

```

```
//*****************************************************************************
// Variables
//*****************************************************************************
uint32_t i = 0;
uint32_t lugares = 0;
uint32_t S1 = 0;
uint32_t S2 = 0;
uint32_t S3 = 0;
uint32_t S4 = 0;
uint8_t ST = 0;

//*****************************************************************************
// Prototipos de Funciones
//*****************************************************************************
void uart_test(void);
void delay(uint32_t msec);
void delay1ms(void);
void Display7(uint32_t numero);

```

```
//*****************************************************************************
// Funcion Principal
//*****************************************************************************
int main(void)
{
    // Se setea osciladon externo de 16MHz
      SysCtlClockSet(SYSCTL_SYSDIV_1 | SYSCTL_USE_OSC | SYSCTL_OSC_MAIN| SYSCTL_XTAL_16MHZ);  //16MHz

   // Se asigna reloj a puerto F, este se encuentra afuera de la tiva.
   SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOB);
   SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOC);
   SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOD);
   SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOE);
   SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);

   // Se configuran los puertos  salidas para el 7 segmentos
   GPIOPinTypeGPIOOutput(GPIO_PORTB_BASE, GPIO_PIN_6|GPIO_PIN_5|GPIO_PIN_4|GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_PIN_0);
   // Se configuran los puertos  entradas los pushbutton y variar la frecuencia.
   GPIOPinTypeGPIOInput(GPIO_PORTC_BASE, GPIO_PIN_4|GPIO_PIN_5|GPIO_PIN_6|GPIO_PIN_7);
   GPIOPadConfigSet(GPIO_PORTC_BASE, GPIO_PIN_4|GPIO_PIN_5|GPIO_PIN_6|GPIO_PIN_7, GPIO_STRENGTH_8MA, GPIO_PIN_TYPE_STD_WPD);
   // Se configuran los puertos  salidas para las leds de salida.
   GPIOPinTypeGPIOOutput(GPIO_PORTD_BASE, GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_PIN_0);
   //configuracion para iniciar la comunicacion UART.
   HWREG(GPIO_PORTD_BASE + GPIO_O_LOCK) = GPIO_LOCK_KEY;
   HWREG(GPIO_PORTD_BASE + GPIO_O_CR) |= GPIO_PIN_7;
   GPIOPinConfigure(GPIO_PD7_U2TX);


   SysCtlPeripheralEnable(SYSCTL_PERIPH_UART2); // habilita el uart2
   GPIOPinTypeUART(GPIO_PORTD_BASE, GPIO_PIN_6 | GPIO_PIN_7); // pines de control del uart
   UARTConfigSetExpClk(UART2_BASE, SysCtlClockGet(), 115200, (UART_CONFIG_WLEN_8 | UART_CONFIG_STOP_ONE | UART_CONFIG_PAR_NONE));
   UARTIntClear(UART2_BASE, UART_INT_RX | UART_INT_RT | UART_INT_TX | UART_INT_FE | UART_INT_PE | UART_INT_BE | UART_INT_OE | UART_INT_RI | UART_INT_CTS | UART_INT_DCD | UART_INT_DSR);
```

```
//*****************************************************************************
// Loop Principal
//*****************************************************************************

   while (1)
   {
       S1 = (GPIOPinRead(GPIO_PORTC_BASE, GPIO_PIN_4) && GPIO_PIN_4);
       S2 = (GPIOPinRead(GPIO_PORTC_BASE, GPIO_PIN_5) && GPIO_PIN_5);
       S3 = (GPIOPinRead(GPIO_PORTC_BASE, GPIO_PIN_6) && GPIO_PIN_6); //lectura de los puertos donde estan los pushbuttons.
       S4 = (GPIOPinRead(GPIO_PORTC_BASE, GPIO_PIN_7) && GPIO_PIN_7);

       ST = GPIOPinRead(GPIO_PORTC_BASE, GPIO_PIN_4|GPIO_PIN_5|GPIO_PIN_6|GPIO_PIN_7) >> 4; //deplazamiento de las variables.

       lugares = ((((4 - S4) - S3) - S2) - S1) ; //suma de los lugares disponibles.
       Display7(lugares);

       GPIOPinWrite(GPIO_PORTD_BASE, GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_PIN_0, ST); //impresion en los leds.

       //funcion que envia los datos al ESP32.
       UARTCharPut(UART2_BASE, S1);
       UARTCharPut(UART2_BASE, S2);
       UARTCharPut(UART2_BASE, S3);
       UARTCharPut(UART2_BASE, S4);
       delay(100);
   }

}
```

```
//*****************************************************************************************************************
// Funcion para hacer delay en milisegundos
//*****************************************************************************************************************
void delay(uint32_t msec)
{
    for (i = 0; i < msec; i++)
    {
        delay1ms();
    }

}
//*****************************************************************************************************************
// Funcion para hacer delay de 1 milisegundos
//*****************************************************************************************************************
void delay1ms(void)
{
    SysTickDisable();
    SysTickPeriodSet(16000);
    SysTickEnable();

    while ((NVIC_ST_CTRL_R & NVIC_ST_CTRL_COUNT) == 0); //Pg. 138
}

void Display7(uint32_t numero)
{
    switch(numero)
    {
    case 0: GPIOPinWrite(GPIO_PORTB_BASE, GPIO_PIN_6|GPIO_PIN_5|GPIO_PIN_4|GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_PIN_0, 0x77); break;
    case 1: GPIOPinWrite(GPIO_PORTB_BASE, GPIO_PIN_6|GPIO_PIN_5|GPIO_PIN_4|GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_PIN_0, 0x14); break;
    case 2: GPIOPinWrite(GPIO_PORTB_BASE, GPIO_PIN_6|GPIO_PIN_5|GPIO_PIN_4|GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_PIN_0, 0x3B); break;
    case 3: GPIOPinWrite(GPIO_PORTB_BASE, GPIO_PIN_6|GPIO_PIN_5|GPIO_PIN_4|GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_PIN_0, 0x3E); break;
    case 4: GPIOPinWrite(GPIO_PORTB_BASE, GPIO_PIN_6|GPIO_PIN_5|GPIO_PIN_4|GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_PIN_0, 0x5C); break;
    default: break;
    }
}
```

Código en la ESP32:

```
//*****************************************************************************************************************
//  Alumno: Mariandree Rivera
//  Carnet: 18178
//  Proyecto # 4, Parqueo
//*****************************************************************************************************************

//*****************************************************************************************************************
// Librerías
//*****************************************************************************************************************
#include <WiFi.h>
#include <WebServer.h>
//*****************************************************************************************************************
// Variables globales
//*****************************************************************************************************************
// SSID & Password
const char* ssid = "TIGO-ACF6";  // Enter your SSID here
const char* password = "2NB123204022";  //Enter your Password here

uint8_t S[4];
uint8_t n_P = 0;
WebServer server(80);  // Object of WebServer(HTTP port, 80 is defult)

uint8_t LED1pin = 2;
bool LED1status = LOW;
```

```
//*************************************************************************************************************
// Configuración
//*************************************************************************************************************
void setup() {
  Serial.begin(115200);
  Serial2.begin(115200);
  Serial.println("Try Connecting to ");
  Serial.println(ssid);

  pinMode(LED1pin, OUTPUT);

  // Connect to your wi-fi modem
  WiFi.begin(ssid, password);

  // Check wi-fi is connected to wi-fi network
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected successfully");
  Serial.print("Got IP: ");
  Serial.println(WiFi.localIP());  //Show ESP32 IP on serial

  server.on("/", handle_OnConnect); // Directamente desde e.g. 192.168.0.15
  server.on("/leer", handle_Data); // handler para enviar datos


  server.onNotFound(handle_NotFound);

  server.begin();
  Serial.println("HTTP server started");
  delay(100);
}
//*************************************************************************************************************
// loop principal
//*************************************************************************************************************
void loop() {
  server.handleClient();          //Inicia servidor
  if (Serial2.available())        //Inicia lectura del UART
    {
      digitalWrite(2, 1);         //Enciende led siempre que se reciba datos por UART
        for (int i = 0; i<=3; i++)//se lee 4 bytes y se almacenan en un array
        {
          S[i] = Serial2.read();

        }
        n_P = (((4 - S[0])- S[1])- S[2])- S[3];  //Se realiza el calculo de cuantos parqueos hay

    }
    digitalWrite(2, 0);         //apaga led de lectura UART


}
```

```
//**************************************************************************************************************
// Handler de Inicio página
//**************************************************************************************************************
void handle_OnConnect() {
  server.send(200, "text/html", SendHTML());
}
//**************************************************************************************************************
// Handlers
//**************************************************************************************************************
void handle_Data() {                          //se envia la data y se concatena los datos a enviar
   String dato_json = "{\"parqueol especial\":";
          dato_json += S[0];
          dato_json += ",";
          dato_json += "\"parqueo2\":";
          dato_json += S[1];
          dato_json += ",";
          dato_json += "\"parqueo3\":";
          dato_json += S[2];
          dato_json += ",";
          dato_json += "\"parqueo4\":";
          dato_json += S[3];
          dato_json += ",";
          dato_json += "\"lugares\":";
          dato_json += n_P;
          dato_json += "}";
  server.send(200, "application/json", dato_json);
}


//**************************************************************************************************************
// Procesador de HTML
//**************************************************************************************************************
String SendHTML(void) {
  String ptr = "<!DOCTYPE html>\n";
  ptr = "<html>\n";
  ptr += "<head>\n";
  ptr += "<style>\n";
  ptr += "body {font-family: Helvetica, sans-serif;}\n";
  ptr += "</style>\n";
  ptr += "</head>\n";
  ptr += "<body>\n";
  ptr += "<h1>Proyecto 4: Drubi Parqueos</h1>\n";
  ptr += "<h3>Mariandree Rivera 18178</h3>\n";
  ptr += "<canvas id=\"Parqueo 1\" width=\"200\" height=\"200\" style=\"border:0px solid #000000;\">\n";
  ptr += "</canvas>\n";
  ptr += "<canvas id=\"Parqueo 2\" width=\"200\" height=\"200\" style=\"border:0px solid #000000;\">\n";
  ptr += "</canvas>\n";
  ptr += "<canvas id=\"Parqueo 3\" width=\"200\" height=\"200\" style=\"border:0px solid #000000;\">\n";
  ptr += "</canvas>\n";
  ptr += "<canvas id=\"Parqueo 4\" width=\"200\" height=\"200\" style=\"border:0px solid #000000;\">\n";
  ptr += "</canvas>\n";
  ptr += "<canvas id=\"Cantidad\" width=\"300\" height=\"40\" style=\"border:0px solid #000000;\">\n";
  ptr += "</canvas>\n";
  ptr += "<script>\n";
```

```
ptr += "function Parqueos(n_parqueo, valor){\n";
ptr += "var canvas = document.getElementById(n_parqueo);\n";
ptr += "var ctx = canvas.getContext(\"2d\");\n";
ptr += "if (valor == 0){\n";
ptr += "ctx.fillStyle = \"#298f0a\";\n";
ptr += "};\n";
ptr += "if (valor == 1){\n";
ptr += "ctx.fillStyle = \"#ff0000\";\n";
ptr += "};\n";
ptr += "ctx.fillRect(0,0,200,100);\n";
ptr += "ctx.fillStyle = \"#000000\";\n";
ptr += "ctx.font = \"20px Arial\";\n";
ptr += "ctx.fillText(n_parqueo, 30, 60);\n";
ptr += "};\n";


ptr += "function Numeros(cantidad){\n";
ptr += "var canvas = document.getElementById(\"Cantidad\");\n";
ptr += "var ctx = canvas.getContext(\"2d\");\n";
ptr += "ctx.fillStyle = \"#00ffae\";\n";
ptr += "ctx.fillRect(0,0,825,40);\n";
ptr += "ctx.fillStyle = \"#000000\";\n";
ptr += "ctx.font = \"30px Arial\";\n";
ptr += "ctx.fillText(\"Lugares disponibles: \" + cantidad.toString(), 0,30);\n";
ptr += "};\n";


ptr += "var sendHttpRequest = function(){\n";
ptr += "var xhr = new XMLHttpRequest();\n";
ptr += "xhr.open(\"GET\", \"http://192.168.0.15/leer\");\n";
ptr += "xhr.responseType = \'json\';\n";
//ptr += "var receive = xhr.responseText;\n";
ptr += "xhr.onload = function() {\n";
ptr += "  console.log(xhr.response);\n";
ptr += "Numeros(xhr.response.lugares);\n";
ptr += "Parqueos(\"Parqueo 1\", xhr.response.parqueo1);\n";
ptr += "Parqueos(\"Parqueo 2\", xhr.response.parqueo2);\n";
ptr += "Parqueos(\"Parqueo 3\", xhr.response.parqueo3);\n";
ptr += "Parqueos(\"Parqueo 4\", xhr.response.parqueo4);\n";
ptr += "};\n";
ptr += "xhr.send();\n";
ptr += "return xhr.response;\n";
ptr += "};\n";
ptr += "setInterval(function(){\n";
ptr += "sendHttpRequest();\n";
ptr += "},1);\n";
ptr += "</script>\n";
ptr += "</body>\n";
ptr += "</html>\n";
return ptr;
```

```
//*************************************************************************************************************
// Handler de not found
//*************************************************************************************************************
void handle_NotFound() {
  server.send(404, "text/plain", "Not found");
}
```

Link:

https://youtu.be/0onvoNfAUgY