floodDetector | Arduino IDE 2.0.3

File Edit Sketch Tools Help

DOIT ESP32 DEVKIT V1

floodDetector.ino

```
/*
floodDetecter.ino
Author - Hashir Sibtain
This program reads water level in a container, displays the reading on an OLED display and sends it to a ThingSpeak channel.
If the water level rises above a certain thershold a warning is displayed on the OLED screen in bold font and the buzzer is
sounded to attract attention.
*/

#include <Arduino.h> // Arduino.h contains many commonly used definitions
#include "WiFi.h" // WiFi.h is included to connect to WiFi
#include <ThingSpeak.h> // ThingSpeak library is the official library to connect to a ThingSpeak channel

// Below libraries are required for interfacing SSD1306 OLED display with the microcontroller
#include <Wire.h> // The Wire library is used for I2C communication with the OLED display
#include <Adafruit_GFX.h> // Adafruit_GFX library is a low level library which is used as a base by the Adafruit_SSD1306 library for the OLED display
#include <Adafruit_SSD1306.h> //Contains all the functions and definitions to interface with the SSD1306 OLED display conveniently

// NewPing is a HC-SR04 Ultrasonic Sensor library which makes taking readings easy and convenient as it handles all the
// calculations and formulae.
#include <NewPing.h>

// We define the trigger and echo pins for the ultrasonic sensor and also define a maximum distance which it can measure
#define TRIG_PIN 5
#define ECHO_PIN 18
#define MAX_DIST 2000

// In the below two lines, we define the width and height of the OLED display in pixels
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64

// We define the WiFi network name, password and timeout in millisecond
#define WIFI_NETWORK "Your WiFi Network" // For privacy and security, this particular WiFi network is only a placeholder and should be replaced with the user's WiFi new
#define WIFI_PASSWORD "Your WiFi Password" // For privacy and security, this particular WiFi network password is only a placeholder and should be replaced with the user
```

Output

Ln 12, Col 1 UTF-8 DOIT ESP32 DEVKIT V1 [not connected]

floodDetector | Arduino IDE 2.0.3

File Edit Sketch Tools Help

DOIT ESP32 DEVKIT V1

floodDetector.ino

```
// We define the WiFi network name, password and timeout in millisecond
#define WIFI_NETWORK "Your WiFi Network" // For privacy and security, this particular WiFi network is only a placeholder and should be replaced with the user's WiFi new
#define WIFI_PASSWORD "Your WiFi Password" // For privacy and security, this particular WiFi network password is only a placeholder and should be replaced with the user
#define WIFI_TIMEOUT 20000 // The ESP 32 will try to connect to the WiFi network for 20 seconds before moving on

const char* writeAPIKey = "0123456789ABCDEF"; // The ThingSpeak Write API Key is required to write to a ThingSpeak Channel. For privacy and security, this is a dummy ke
unsigned long channelNumber = 1234567; // The ThingSpeak Channel ID identifies a unique ThingSpeak Channel so the data reaches the correct channel. For privacy and secu

unsigned long lastTime = 0; // long variable to keep time
int upDelay = 5000; // Every 5 seconds, the ESP32 will try uploading to the ThingSpeak channel
NewPing levelSensor(TRIG_PIN, ECHO_PIN, MAX_DIST); // We create a NewPing variable which represents the HC-SR04 Ultrasonic Sensor

#define OLED_RESET -1 // Reset pin related to OLED Display
#define SCREEN_ADDRESS 0x3C // OLED Display's I2C Slave address
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET); // We create a new display variable which represents the SSD1306 OLED Display

const int buzzer = 4; // This constant integer denotes the pin the buzzer is connected to

WiFiClient client; // We create a WiFi Client for use with the ThingSpeak library

void connectToWiFi()
{
  /* This is a user defined function which connects the ESP32 to WiFi and initializes the ThingSpeak library.
     It returns nothing and takes no arguments
  */
  Serial.print("Connecting to WiFi"); // Print a line denoting the start of an attempt to connect to WiFi
  WiFi.mode(WIFI_STA); // Set ESP32 WiFi mode to station mode to connect to a WiFi hotspot
  ThingSpeak.begin(client); // Initialize the ThingSpeak library and network setting with the WiFi client
  WiFi.begin(WIFI_NETWORK, WIFI_PASSWORD); // Connect the ESP32 to a WiFi network using the provided credentials

  unsigned long startAttemptTime = millis(); // This variable keeps track of time while the ESP attempts connecting to the WiFi network
  while(WiFi.status() != WL_CONNECTED && millis() - startAttemptTime < WIFI_TIMEOUT) // This while loop prints a '.' while the ESP is attempting to connect to WiFi.
  {                                                                                  // It ends when the connection is successful or when WIFI_TIMEOUT milliseconds (in
```

Output

indexing: 39/49 Ln 12, Col 1 UTF-8 DOIT ESP32 DEVKIT V1 [not connected]

floodDetector | Arduino IDE 2.0.3

File Edit Sketch Tools Help

DOIT ESP32 DEVKIT V1

floodDetector.ino

```
  unsigned long startAttemptTime = millis(); // This variable keeps track of time while the ESP attempts connecting to the WiFi network
  while(WiFi.status() != WL_CONNECTED && millis() - startAttemptTime < WIFI_TIMEOUT) // This while loop prints a '.' while the ESP is attempting to connect to WiFi.
  {                                                                                   // It ends when the connection is successful or when WIFI_TIMEOUT milliseconds (in
    Serial.print("."); // prints the '.;
    delay(100); // delay for 100 ms
  }

  if(WiFi.status() != WL_CONNECTED) // Print 'Failed if WiFi not connected
  {
    Serial.println("Failed.");
  }
  else // else print connected successfully and print local IP
  {
    Serial.println("Connected successfully");
    Serial.println(WiFi.localIP());
  }
}

void warn(int waterLevel)
{
  /* This function is user-defined and deals with the display of the high water-level warning on the
  SSD1306 OLED display. It does various things like set cursor to a correct position, set an appropriate
  font size and an appropriate colour and finally it displays the warning which consists of the word 'WARNING'
  written in large font with white background and black text on it and the water level on the next line
  in bold font */
  display.setCursor(11, 16); // Set the cursor position to 11 pixels from the right (x co-ordinate) and 16 pixels from the top (y co-ordinate)
  display.clearDisplay(); // Clear the display to get a fresh display and space to write on
  display.setTextSize(2); // Set text size to 2
  display.setTextColor(BLACK, WHITE); // Set text colour to black and background colour to white where 'BLACK' and 'WHITE' are macros defined in Adafruit_SSD1306.h file
  display.println(" WARNING "); // Display the text 'WARNING' as specified above
  display.setTextColor(WHITE); // Set text colour to white and display colour to black
  display.setCursor(50, 40); // Set the cursor position to 50 pixels from the right (x co-ordinate) and 40 pixels from the top (y co-ordinate)
  display.print(waterLevel); // Display the water level as specified by the above two lines
```

Output

Ln 12, Col 1 UTF-8 DOIT ESP32 DEVKIT V1 [not connected]

floodDetector | Arduino IDE 2.0.3

File Edit Sketch Tools Help

DOIT ESP32 DEVKIT V1

floodDetector.ino

```
  SSD1306 OLED display. It does various things like set cursor to a correct position, set an appropriate
  font size and an appropriate colour and finally it displays the warning which consists of the word 'WARNING'
  written in large font with white background and black text on it and the water level on the next line
  in bold font */
  display.setCursor(11, 16); // Set the cursor position to 11 pixels from the right (x co-ordinate) and 16 pixels from the top (y co-ordinate)
  display.clearDisplay(); // Clear the display to get a fresh display and space to write on
  display.setTextSize(2); // Set text size to 2
  display.setTextColor(BLACK, WHITE); // Set text colour to black and background colour to white where 'BLACK' and 'WHITE' are macros defined in Adafruit_SSD1306.h file
  display.println(" WARNING "); // Display the text 'WARNING' as specified above
  display.setTextColor(WHITE); // Set text colour to white and display colour to black
  display.setCursor(50, 40); // Set the cursor position to 50 pixels from the right (x co-ordinate) and 40 pixels from the top (y co-ordinate)
  display.print(waterLevel); // Display the water level as specified by the above two lines
  display.print("cm"); // Display the unit 'cm' with the water level
  display.display(); // Actually draw on the OLED display

}

void buzz()
{
  /* This function is concerned with sounding the buzzer and returns nothing. It sounds the alarm
     according to a particular pattern */
  tone(buzzer, 1000); // the tone() function plays a particular takes two parameters, the buzzer pin and the frequency. It plays a tone of that frequency on the buzzer.
  delay(500); // The program waits for 0.5 seconds. So, for these 0.5 seconds the buzzer keeps playing the 1 KHz tone
  noTone(buzzer); // the noTone() function takes only one argument, the buzzer pin and stops any tone that is playing on that buzzer. So, the 1KHz tone stops
}
void setup() {
  // In this section we setup Serial communication, WiFi, and OLED display
  Serial.begin(115200); // we begin serial communication at 115200 bps
  connectToWiFi(); // we call the connectToWiFi() function

  if(!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) // we try initializing the SSD1306 OLED display module. if an error occurs we print 'SSD1306 allocation failed
  {
    Serial.println(F("SSD1306 allocation failed"));
```

Output

Ln 12, Col 1 UTF-8 DOIT ESP32 DEVKIT V1 [not connected]

floodDetector | Arduino IDE 2.0.3

File Edit Sketch Tools Help

DOIT ESP32 DEVKIT V1

floodDetector.ino

```
  if(!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) // we try initializing the SSD1306 OLED display module. if an error occurs we print 'SSD1306 allocation faile
  {
    Serial.println(F("SSD1306 allocation failed"));
    while(true);
  }
  // The following four lines are related to OLED display module
  display.clearDisplay(); // clear the display
  display.setTextSize(1); // set font size to 1
  display.setTextColor(WHITE); // set font colour to White
  display.setCursor(0,28); // set cursor position to co ordinate (0, 28)

}

void loop() {
  // This is the main loop of the program where the code repeatedly runs. It takes readings, sends them to ThingSpeak, display them on OLED screen and sound the buzzer
  delay(50); // small delay for callibration

  int lev = 22 - levelSensor.ping_cm(); // lev keeps track of the water level. The levelSensor.ping_cm() function is from the NewPing library and returns reading
                                        // from the HC-SR04 sensor in cm. Since, the initial distance of the sensor from the bottom of the container is 22cm, we
                                        // subtract the reading from 22 to get the height of the water.

  if((millis() - lastTime) > upDelay)   // If, upDelay amount of time or more has passed, then send the data to the ThingSpeak channel.
  {
    lastTime = millis(); // update the time in tracking variable
    Serial.println("Uploading to ThingSpeak"); // print "Uploading to ThingSpeak" on serial monitor
    int x = ThingSpeak.writeField(channelNumber, 1, lev, writeAPIKey); // Actually attempt uploading the data to the ThingSpeak channel using given credentials. It retu
    if (x == 200) // If x is 200, upload was successfull so print a success message on serial monitor
      Serial.println("Successfully uploaded to ThingSpeak!");
    else Serial.println(x); // else print out the error code
  }
  if(lev > 10) // If lev is greater than 10 cm then display the warning on OLED display by calling the warn() function and sound the buzzer by calling the buzz() functi
  {
```

Output

Ln 12, Col 1 UTF-8 DOIT ESP32 DEVKIT V1 [not connected]

floodDetector | Arduino IDE 2.0.3

File Edit Sketch Tools Help

DOIT ESP32 DEVKIT V1

floodDetector.ino

```
                                        // from the HC-SR04 sensor in cm. Since, the initial distance of the sensor from the bottom of the container is 22cm, we
                                        // subtract the reading from 22 to get the height of the water.

  if((millis() - lastTime) > upDelay)   // If, upDelay amount of time or more has passed, then send the data to the ThingSpeak channel.
  {
    lastTime = millis(); // update the time in tracking variable
    Serial.println("Uploading to ThingSpeak"); // print "Uploading to ThingSpeak" on serial monitor
    int x = ThingSpeak.writeField(channelNumber, 1, lev, writeAPIKey); // Actually attempt uploading the data to the ThingSpeak channel using given credentials. It retu
    if (x == 200) // If x is 200, upload was successfull so print a success message on serial monitor
      Serial.println("Successfully uploaded to ThingSpeak!");
    else Serial.println(x); // else print out the error code
  }
  if(lev > 10) // If lev is greater than 10 cm then display the warning on OLED display by calling the warn() function and sound the buzzer by calling the buzz() functi
  {
    warn(lev);
    buzz();
  }
  else // Else, print the water level normally on OLED display
  {
    display.clearDisplay(); // clear display
    display.setTextSize(1); // set font size to 1
    display.setTextColor(WHITE); // set font colour to White
    display.setCursor(5,28); // set cursor position to co-ordinate (5,28)
    display.clearDisplay(); // clear display
    display.print("Water Level: "); // print "Water Level: " at the configured co-ordinate and font size and style on the OLED display
    display.print(lev); // print the actual water level on the OLED display
    display.print("cm"); // print "cm" on the OLED display
    display.display(); // finally, display the text on the OLED display so that everything shows up
  }


}
```

Output

Ln 12, Col 1 UTF-8 DOIT ESP32 DEVKIT V1 [not connected]