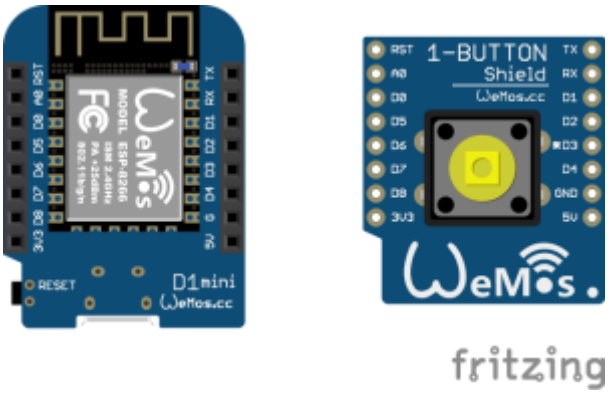
wemos, telegram, botón

Botón de emergencia con Wemos

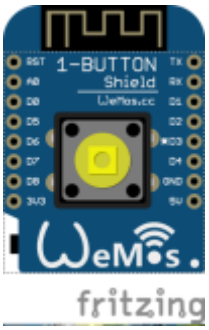
Después de leer el artículo de **jeanotP1314** sobre como construir un botón de ayuda para personas mayores, decidí hacer algo similar pero utilizando telegram en vez de gmail y añadiendole la posibilidad de recibir o enviar comandos para ejecutar acciones.

De momento sólo tengo implementado algunos comandos, pero la lista de acciones se puede ampliar muy fácilmente para leer o interactuar con otros sensores conectados al Wemos.

Para realizar este proyecto sólo necesitamos un Wemos y un Button Shield



Simplemente ponemos el button Shield encima del Wemos verificando que coinciden las patillas de cada módulo.



Lo primero como siempre es crear nuestro bot en telegram para poder enviar y recibir los mensajes. Si no sabes como hacerlo está explicado aquí

Alimentamos el Wemos y cargamos el siguiente código:

```
/* Código original de la conexión con telegram de  https:
//github.com/gusman126/arduino_telegram_bot
  Modificado por LC (wiki.intrusos.info)
*/
#include <WiFiClientSecure.h>
#include <ESP8266WiFi.h>
#include "DHT.h"
```

```
// Iniciamos la conexión con el router
const char *ssid = "ssid de la wifi";   // no puede ser mayor de 32
caracteres
const char *pass = "contraseña de la wifi";   // la contraseña de la WIFI
int status = WL_IDLE_STATUS;

// Datos del Bot de Telegram
String BOTtoken = "bot26621579:AAE1X6V2MHVxOVC-iK_a_4dUi6f0Xf603vh";
//cambiar por el token de tu bot
String Chat_id = "205215515"; // cambiar por tu Chat_id,
String Texto_enviar = "";
String Texto_recibido = "";
String Update_id = "";
String anterior_upd = "";
String Nueva_upd = "";
String Respuesta = "";

// Variables de tiempo
int Inicio;
int Termino;
int Intervalo = 15000;
unsigned long elapsed = 0;
unsigned long previous;
boolean respondio = false;

//variables de estado
int estadoboton = 0;

// Pin del wemos al que está conectado el botón
const int botonpin = D3;
const int ledpin = BUILTIN_LED;

WiFiClientSecure client; // inicio del cliente seguro
IPAddress server(149, 154, 167, 200); // IP de api.telegram.org telegram


void setup() {

  Serial.begin(115200);
  WiFi.begin(ssid, pass); // Conexion a Wifi

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("Conectado a la red WiFi");
  Serial.println("Dirección IP: ");
  Serial.println(WiFi.localIP());

  // Comprobamos la conexión a Telegram
```

```
  if (client.connect(server, 443)) {
    Serial.println(".... conectado a Telegram");
  }
  // y enviamos el texto de inicio
  Enviar_texto(" Inicio del Sistema .......");

  //Comprobamos el último mensaje
  Ultimo_msg();
  previous = millis();

  // inicializamos el led pin
  pinMode(ledpin, OUTPUT);

  // inicializamos el botón pin
  pinMode  (botonpin, INPUT);

  //inicializamos el led
  digitalWrite(ledpin, HIGH); //Con HIGH el led está apagado

  //Inicializamos el buzzer
  pinMode (5, OUTPUT);   //es la conexión D1->GPIO5

}


void loop() {
  estadoboton = digitalRead(botonpin);

  if (estadoboton == LOW) {            //Cuando presionamos el botón
    digitalWrite(ledpin, LOW);
    Enviar_texto("Necesito Ayuda");
    Serial.println("Necesito ayuda");
  }

  while (estadoboton == LOW ) {  //Una vez pulsado el botón está a la espera
de nuestros mensajes

    elapse();
    Leer_msg(); // leemos el último mensaje


    // Comprobamos que haya pasado xx seg desde la última vez
    if (elapsed > 500) {
    }
    anterior_upd = Update_id; // Guardamos la anterior Update
    Ultimo_msg (); // comprobamos el último mensaje
    delay(1000); // Esperamos a recibir los datos
    Leer_msg(); // Leemos los datos
    busca_upd_id(Respuesta); // buscamos la Update_id y la guardamos
    busca_texto(Respuesta); // Buscamos el Texto del mensaje
```

```
    // Si ha cambiado la Update_id seguimos con el código
    if (anterior_upd != Nueva_upd) {
      Responder_mensaje(Texto_recibido);
    } else {
    } // No hacemos nada si es el mismo Upd_id
  }
} // Fin Loop


// Orden para buscar el texto del mensaje
void busca_texto( String Rsp ) {
  Texto_recibido = "";
  int  start = Rsp.indexOf("text") + 7 ; // Buscamos el índice ( número ) de
la palabra "text" y le añadimos 7
  int  fin = Rsp.indexOf("}}]}") - 1; // Buscamos el índice del texto }}]} y
le restamos uno
  Texto_recibido = (Rsp.substring(start, fin)); // Guardamos el resultado en
la variable
}

//Orden para buscar la Update_id
void busca_upd_id( String Rsp ) {
  anterior_upd = Update_id; // Guardamos la anterior Update_id para
comprobar
  int  start = Rsp.indexOf("update_id") + 11 ; // Buscamos el índice del
texto y le añadimos 11
  int  fin = Rsp.indexOf("message") - 2; // Buscamos el índice del texto y
le restamos 2
  Update_id = Rsp.substring(start, fin); // Guardamos la Update_id
  Nueva_upd = Rsp.substring(start, fin); // Volvemos a guardar la Update_id
pero en la variable de nueva
}

// Orden para pedir el ultimo mensaje, vemos que se usa el Offset=-1&limit=1
para mostrar sólo el último
void Ultimo_msg () {
  if (client.connect(server, 443)) {
    // client.println("GET /botxxxx/getUpdates?offset=-1&limit=1");
    client.println("GET /" + BOTtoken + "/getUpdates?offset=-1&limit=1");
  }
  previous = millis(); // Guardamos los milisegundos para comprobar que haya
pasado X tiempo entre lecturas
}

//Leemos el mensaje completo y lo añadimos a una variable carácter por
carácter
void Leer_msg () {
  Respuesta = ""; // Vaciamos la variable
  while (client.available()) { // Mientras no lo lea todo seguirá leyendo
    char inChar = client.read(); // Lee el carácter
    Respuesta += inChar; // Añadimos carácter a carácter el mensaje
```

```
  }
}

//Orden para comprobar el tiempo entre lecturas
void elapse() {
  elapsed = millis() - previous;
}


// Función para hacer sonar el Buzzer
  void beep(unsigned char pausa)
        {
            analogWrite(5, 20);
            delay(pausa);                // Espera
            analogWrite(5, 0);           // Apaga
            delay(pausa);                // Espera
        }

//Orden para enviar cualquier texto a Telegram
void Enviar_texto( String Texto_enviar ) {
  if (client.connect(server, 443)) {
    client.println("GET /" + BOTtoken + "/sendMessage?chat_id=" + Chat_id +
"&text=" + Texto_enviar + "");
    Serial.println("GET /" + BOTtoken + "/sendMessage?chat_id=" + Chat_id +
"&text=" + Texto_enviar + "");
  }
}

//Aquí añadiremos las ordenes de respuesta del arduino
void Responder_mensaje ( String mensaje ) {

  if (mensaje == "Gps") {
    Enviar_texto("Las coordenadas son");
    respondio = true;
  }
  else if (mensaje == "Alarma") {
    Enviar_texto("Avisando");
    beep(500);
    respondio = true;
  }
  else if (mensaje == "Cerrar") {
    Enviar_texto("Aviso terminado");
    Serial.println("Aviso terminado");
    respondio = true;
    digitalWrite(ledpin, HIGH);
    estadoboton = 1;
  }

  if (respondio == true) { // mostramos el texto que se ha entendido
    Serial.println("El Texto : " + mensaje + " Lo he entendido
perfectamente");
```

```
  }
  else {
    Serial.println("El Texto : " + mensaje + " No lo he entendido");

  }
  respondio = false ; // Dejamos en falso que entendió el mensaje
}

////////// Fin del código
```

En este ejemplo cuando se presiona el botón se envía un mensaje pidiendo ayuda y se entra en un bucle para ejecutar los comandos que le enviamos por mensajes de telegram hasta que le llegue un mensaje de **Cerrar** .

Para hacer pruebas he conectado un buzzer al PIN D1 (GPIO5) para hacerlo sonar como aviso y en cuanto tenga en mi poder el módulo de GPS y un transceiver con **Lora** intentaré ampliar el código de ejemplo, para obtener las coordenadas de posición y a su vez utilizar Lora en vez de una conexión Wifi.