Workshop Arduino and ESPNOW

Topics:

- Arduino IDE installation
- ESP8266 card installation
- Drivers CH340
- Blink code
- Programming with Arduino
- FastLed library
- ESP-now
- Comments

Install Arduino

Install Arduino IDE on your computer is a simple process. Below I provide you with the steps to install Arduino IDE on Windows, macOS and Linux systems:

For Windows:

1. **Download Arduino IDE**
   - Do click in the link https://downloads.arduino.cc/arduino-1.8.19-windows.exe
2. **Run the Installer**:
   - Once the download is complete, run the installation file (usually it has an .exe extension).
   - The system may ask you for administrator permissions to continue. Accept these permissions.
3. **Install Arduino IDE**:
   - Follow the instructions of the installation wizard. You can choose to change the installation location or simply follow the default settings.
   - Installation should be quick and easy.
4. **Finish Installation**:
   - Once the installation is complete, you should see a message informing you that the Arduino IDE has been installed successfully.
   - You can run Arduino IDE from the start menu.

For macOS:

1. **Download Arduino IDE**:
   - Do click in the linkhttps://downloads.arduino.cc/arduino-1.8.19-macosx.zip
2. **Mount the DMG File**:
   - Open the downloaded DMG file.
   - Drag the Arduino application to the Applications folder.
3. **Run Arduino IDE**:
   - Open the Applications folder and search for the Arduino application.
   - Double click on Arduino to run it.

For Linux (Free):

1. **Install from the Repository (Recommended):**

  - Open a terminal.
  - Run the following command to install Arduino IDE:

    ```
    sudo apt-get update
    sudo apt-get install arduino
    ```

2. **Run Arduino IDE**:
  - Once installed, search for "Arduino IDE" in the applications menu or run "arduino" in the terminal to open the IDE.

After following these steps, you will have the Arduino IDE installed on your computer and can start programming your projects with Arduino.

Install ESP8266 on Arduino IDE

The ESP8266 is not included in the Arduino IDE by default, but we can easily add it using the “Card Manager”.

Open the Arduino IDE and go to “File” > “Preferences”. In the “Additional card manager URLs” field, add the following URL:

http://arduino.esp8266.com/stable/package_esp8266com_index.json

Then, we go to “Tools” > “Board” > “Card Manager”.

In the panel that appears, we search for “ESP8266” in the search bar, and click on “Install” in the package that appears.

Once the installation is complete, you can choose the ESP8266 models in the drop-down menu, and you will be ready to program the ESP8266.

Drivers CH340

To program Arduino boards that use the CH340 chip, you need to install the CH340 driver on your computer. Below I provide you with the steps to install the CH340 driver on Windows, macOS and Linux systems:

For Windows:

1. **Connect the Arduino Board**:
    - Connect your Arduino board to the USB port on your computer.
2. **Download CH340 Driver**:
    - https://sparks.gogo.co.nz/assets/_site_/downloads/CH34x_Install_Windows_v3_4.zip
    - https://www.arduined.eu/files/windows10/CH341SER.zip
3. **Run the Installer**:
    - Unzip the downloaded file if necessary.
    - Run the installation file (usually it has an .exe extension).
    - Follow the instructions of the installation wizard.
4. **Finish Installation**:
    - Once the installation is complete, you should see a confirmation message.
    - Your Arduino board with CH340 chip should now be recognized by Windows.

For macOS:

1. **Connect the Arduino Board**:
    - Connect your Arduino board to the USB port on your computer.
2. **Installation Automation**:
    - On macOS, you generally do not need to manually install the CH340 driver. The system should automatically recognize the card and configure the necessary driver. But you can download it from the following link:

https://cdn.sparkfun.com/assets/learn_tutorials/8/4/4/CH341SER_MAC.ZIP?_gl=1*1fbryfg*_ga*MjA1NDk1ODQzNS4xNjk3NzE5OTEz*_ga_T369JS7J9N*MTY5NzcxOTkxMy4xLjEuMTY5NzcyMDA1NC42MC4wLjA.

3. **Verify Connection**:
    - Open the Arduino IDE application.
    - In the "Tools" menu, select the model of your Arduino board.

    - Make sure you select the correct serial port in the "Port" menu (e.g. /dev/cu.usbserial-XXXX).

For Linux (Free):

1. **Connect the Arduino Board**:
    - Connect your Arduino board to the USB port on your computer.
2. **Installation Automation**:
    - On many Linux distributions, the CH340 driver is installed automatically without downloading. But you can download it from the following link:

https://cdn.sparkfun.com/assets/learn_tutorials/8/4/4/CH341SER_LINUX.ZIP?_gl=1*v82jlk*_ga*MjA1NDk1ODQzNS4xNjk3NzE5OTEz*_ga_T369JS7J9N*MTY5NzcxOTkxMy4xLjEuMTY5NzcyMDA1NC42MC4wLjA.

3. **Verify Connection**:
    - Open the Arduino IDE application.
    - In the "Tools" menu, select the model of your Arduino board.
    - Make sure you select the correct serial port in the "Port" menu (e.g. /dev/ttyUSB0).

If after following these steps your Arduino board is not recognized, you may need to restart your computer or check for compatibility issues with your operating system version. In most cases, installing the CH340 driver should allow your Arduino board to function properly.

Programming the ESP8266 in Arduino IDE

Let's test that everything works correctly. Connect the ESP8266 to your computer via a USB cable.

In the Arduino IDE, we select the board we are using. Let's go to “Tools” > “Plate” and select “Generic 8266 Module” as the board you are using.

Then we select the COM port to which the ESP8266 is connected in “Tools” > “Port”.

We open the Blink example. Let's go to “File” > “Examples” > “ESP8266” > “Blink”

Library Manager

The library manager allows you to manage the installation and update of Arduino libraries directly from the IDE in a very convenient way. You can find it in Menu / Program / Include Library / Manage libraries

The library manager is very easy to use and we will use it preferably whenever we are missing a library, in case we download someone's project or we are developing our application.

Although it is possible to install libraries in other ways as we will see later, it is always advisable to use the library manager as the first option. This means that you have supported and quality libraries and you do not have to worry about the technical details of their installation. Furthermore, if there are updates in the future, you can always update them with a single click and safely.

How to install a library on Arduino

Let's assume that you want to play with an LED strip and for that you need a library. One is the FastLED library. Since we already know its name, we enter it in the search engine and we will see the latest version available. Let's install it by clicking the Install button.

Activity 1

To light a WS2812 LED using the FastLED library on Arduino, follow these steps:

1. **Hardware Configuration**:
    - Connect the data pin of the WS2812 LED to one of the digital pins on your Arduino board (e.g. Pin D5).
    - Connect the VCC (+5V) pin of the WS2812 LED to the 5V power supply of your Arduino board.
    - Connect the GND (ground) pin of the WS2812 LED to the GND of your Arduino board.
2. **Installation of the Library**:
    - Open the Arduino environment.
    - Go to the "Sketch" menu and select "Include Library" > "Manage Libraries".
    - In the pop-up window, search for "FastLED" and press the "Install" button to install the library in your development environment.

3. **Arduino Code**: Below I provide a simple code example to light a WS2812 LED using FastLED:

```
#include <FastLED.h>

// Pin configuration
#define DATA_PIN 14 // D5 is pin 14

// LED strip configuration
#define NUM_LEDS 1 // Number of LEDs on your strip

CRGB leds[NUM_LEDS]; // An array

void setup() {
  FastLED.addLeds<WS2812, DATA_PIN, GRB>(leds, NUM_LEDS);
}

void loop() {
  // Set the LED color
  leds[0] = CRGB(255, 0, 0); // Bright red
  FastLED.show(); // send data to the LED
  delay(500); // wait 0.5 seconds
  leds[0] = CRGB(0,0,0); // turn off the LED
  FastLED.show(); // send data to the LED
  delay(500);  // wait 0.5 seconds
}
```

4. **Load and Run the Code**:
   - Connect your Arduino board to the computer via a USB cable.
   - Select the board type and port in the Arduino environment.
   - Verify the code and upload it to your Arduino board.

With these steps, the WS2812 LED should turn bright red for half second. You can modify the color and effects according to your preferences by adjusting the values in the code. Have fun experimenting with your WS2812 LEDs!

Activity 2

In the code below we will control the color of the LED sending data to the Serial Output. Let`s try!

```
#include <FastLED.h>

#define NUM_LEDS 1 // number of LED
#define DATA_PIN 14 // D5 is pin 14

int ledcolor;

CRGB leds[NUM_LEDS]; // LED Array

void setup() {
     Serial.begin(115200); // The Serial monitor must be left without line adjustment
     FastLED.addLeds<WS2812, DATA_PIN, GRB>(leds, NUM_LEDS);
}

void loop() {
  if(Serial.available() != 0){
    ledcolor = Serial.parseInt();
    if (ledcolor > 255){
      leds[0] = CHSV(0,0,0);
      FastLED.show(); // send data to the LED
    }else{
      leds[0] = CHSV(ledcolor,255,255);
      FastLED.show(); // send data to the LED
    }
  }
}
```

ESPNOW protocol

ESP-NOW is like a secret code language that two devices (like two walkie-talkies) use to talk to each other.

1. **Secret Code (ESP-NOW Protocol)**: Imagine two friends who want to communicate with each other using a secret code. They

have a set of secret words and meanings that only they understand. In the same way, ESP-NOW is a special language that two electronic devices, like two ESP8266 or ESP32 chips, use to talk to each other.

2. **Two Devices (Transmitter and Receiver)**: Just like you and your friend with walkie-talkies, in ESP-NOW, there are two devices. One is the "transmitter" (like the person speaking into the walkie-talkie), and the other is the "receiver" (like the person listening).
3. **Sending and Receiving Messages**: The transmitter can send messages to the receiver using the ESP-NOW protocol, and the receiver understands these messages because they share the same secret code.
4. **No Internet Needed**: What's cool is that ESP-NOW allows these devices to talk to each other without needing the internet. It's like your walkie-talkies working anywhere, even in places without phone signals.
5. **Short Distances**: ESP-NOW works best over short distances, so it's like when you and your friend can talk to each other within a certain range using your walkie-talkies.
6. **Fast and Reliable**: The ESP-NOW language is very fast and reliable, so the messages are sent and received quickly and accurately, just like when you and your friend communicate using your secret code.

Overall, ESP-NOW is a way for electronic devices to communicate with each other using a special language, and it's really good for making devices like remote-controlled cars or sharing information between gadgets without needing the internet. It's a bit like having your own secret code to chat with your electronic friends!

Activity 3

Below I provide a simple code to get the MAC adress:

```
#include <ESP8266WiFi.h>

void setup(){
  Serial.begin(115200);
  Serial.println();
  Serial.print("ESP8266 Board MAC Address:  ");
  Serial.println(WiFi.macAddress());
}

void loop(){

}
```

Load and Run the Code:

- Connect your Arduino board to the computer via a USB cable.
- Select the board type and port in the Arduino environment.
- Verify the code and upload it to your Arduino board.

With these steps, if you open the Serial Monitor we will see the MAC address for our device. It's unique!

Write it somewhere we will need it later.

Activity 4

For this Activity we need to split in two groups: Senders and Receivers.

Below the code for the Senders: (we need the MAC adress of the receiver device).

```
#include <ESP8266WiFi.h>
#include <espnow.h>

// REPLACE WITH RECEIVER MAC Address ** IMPORTANT**
uint8_t broadcastAddress[] = {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF};

// Structure example to send data
// Must match the receiver structure
typedef struct struct_message {
  int ledcolor;
} struct_message;

// Create a struct_message called myData
struct_message myData;

// Callback when data is sent
void OnDataSent(uint8_t *mac_addr, uint8_t sendStatus) {
  Serial.print("Last Packet Send Status: ");
  if (sendStatus == 0){
    Serial.println("Delivery success");
  }
  else{
    Serial.println("Delivery fail");
  }
}

void setup() {
  // Init Serial Monitor
  Serial.begin(115200);

  // Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);

  // Init ESP-NOW
  if (esp_now_init() != 0) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  // Once ESPNow is successfully Init, we will register for Send CB to
  // get the status of Trasnmitted packet
  esp_now_set_self_role(ESP_NOW_ROLE_CONTROLLER);
  esp_now_register_send_cb(OnDataSent);

  // Register peer
  esp_now_add_peer(broadcastAddress, ESP_NOW_ROLE_SLAVE, 1, NULL, 0);
}

void loop() {
  if(Serial.available() != 0){
    // Set values to send
    myData.ledcolor = Serial.parseInt();
    // Send message via ESP-NOW
    esp_now_send(broadcastAddress, (uint8_t *) &myData, sizeof(myData));
  }
}
```

Below the code for the Receivers:

```
#include <ESP8266WiFi.h>
#include <espnow.h>

#include <FastLED.h>

#define NUM_LEDS 1 // number of LED
#define DATA_PIN 14 // D5 is pin 14

CRGB leds[NUM_LEDS]; // LED Array

// Structure example to receive data
// Must match the sender structure
typedef struct struct_message {
    int ledcolor;
} struct_message;

// Create a struct_message called myData
struct_message myData;

// Callback function that will be executed when data is received
void OnDataRecv(uint8_t * mac, uint8_t *incomingData, uint8_t len) {
  memcpy(&myData, incomingData, sizeof(myData));
  Serial.print("Bytes received: ");
  Serial.println(len);
  Serial.print("Color led: ");
  Serial.println(myData.ledcolor);
  if(myData.ledcolor > 255){
      leds[0] = CHSV(0,0,0);
  }else{
      leds[0] = CHSV(myData.ledcolor,255,255);
  }
  FastLED.show(); // enviamos data al led
}

void setup() {
  // Initialize Serial Monitor
  Serial.begin(115200);

  FastLED.addLeds<WS2812, DATA_PIN, GRB>(leds, NUM_LEDS);

  // Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);

  // Init ESP-NOW
  if (esp_now_init() != 0) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  // Once ESPNow is successfully Init, we will register for recv CB to
  // get recv packer info
  esp_now_set_self_role(ESP_NOW_ROLE_SLAVE);
  esp_now_register_recv_cb(OnDataRecv);
}

void loop() {

}
```

Now we can control the LED of the other Group, let's try!

Comments

Why It's Important to Learn Programming with Arduino:

1. **Building Cool Stuff**: Learning Arduino programming is like having a magic wand to create all sorts of fun gadgets and robots. You can build your own toys, games, and even things that can make your life easier.
2. **Understanding Technology**: We live in a world filled with technology, and knowing how to program with Arduino helps you understand how things work. It's like learning the language of computers, and it can be exciting to control the behavior of machines.
3. **Problem Solving**: When you learn to program, you become a problem solver. You can write instructions to make your creations do what you want. It's like solving puzzles and coming up with solutions.

4. **Creativity**: Arduino programming is like being an artist with technology. You get to be creative and invent your own things. Whether it's making a robot dance or creating a game, the possibilities are endless.

5. **Preparing for the Future**: Learning programming is like learning a superpower for the future. Many jobs and careers involve technology, and understanding programming can open up a world of opportunities.

6. **Fun Learning**: It's not just about studying; it's about having fun while learning. You can experiment, make mistakes, and learn from them. It's like playing a video game where you control the characters.

7. **Sharing with Others**: When you create something cool with Arduino, you can share it with your friends and family. You can even help others learn and create together.

So, learning programming with Arduino is like learning to be a tech magician. You can make your ideas come to life, understand the digital world, and have a lot of fun while doing it. It's not just about today; it's about getting ready for an exciting future filled with technology and innovation!

Thanks! Hope you enjoyed !!! 💡 😃 💻

Sebastian Jara Bunster (c) 2023