# ESP8266, RC522 e MySql: Sistema di validazione remota di un tag RFID

Utilizzando la scheda ESP8266 NodeMCU e l'IDE di Arduino, realizziamo un progetto che permette di leggere un tag RFID e "riconoscerlo" consultando un DataBase MySql su un server remoto.

ESP8266 è interfacciato con un dispositivo RC522 che legge un tag RFID (una carta magnetica o un tag portachiavi). **Il codice letto viene inviato ad un DB MySql remoto**: se è presente nel database, si accende un diodo led verde, viceversa si accende un diodo led rosso.

**Una pagina web**, che si può visualizzare accedendo al server da qualsiasi luogo e con qualsiasi dispositivo, mostra le letture effettuate dal Reader.

Al posto della carta magnetica e del tag portachiavi si può utilizzare un qualsiasi oggetto con tag RFID.
L'accensione del diodo verde simula l'azione che viene intrapresa: l'azionamento di un relé per aprire una porta o per accendere un motore, il tracciamento dell'oggetto, l'aggiornamento del magazzino, il monitoraggio dell'avanzamento della produzione, ecc.

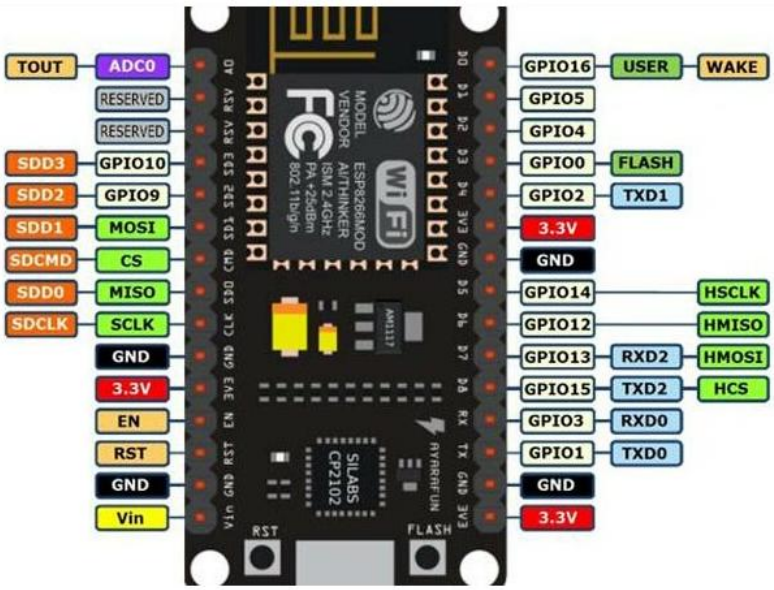
## Premessa

Questo progetto replica nozioni e tecniche già utilizzate in due lavori precedenti pubblicati nella sezione Arduino dell'Area Download del sito www.maurodeberardis.it: Web Server ESP8266 NodeMCU e Arduino e RC522: Lettore di tag RFID.

## Che cos'è l'ESP8266

L'ESP8266 è un SoC (System on a Chip) costituito da un microcontrollore a 32 bit e un ricetrasmettitore Wi-Fi . Così come Arduino, consente di controllare ingressi e uscite e, grazie alle dimensioni ridotte, al basso costo e al modulo Wi-Fi integrato al suo interno, rappresenta una soluzione completa per la maggior parte dei progetti di "Internet of Things" (per connettersi alla rete Wifi e a Internet, per ospitare un Web Server, per controllare un oggetto, ecc.) Inoltre si può programmare facilmente anche con l'IDE di Arduino.

Sul mercato sono disponibili molte schede di sviluppo che utilizzano il chip ESP8266: tra queste ho scelto la scheda **NodeMCU,** forse la più diffusa e popolare sul mercato, con interfaccia USB (convertitore da USB a seriale) integrata.

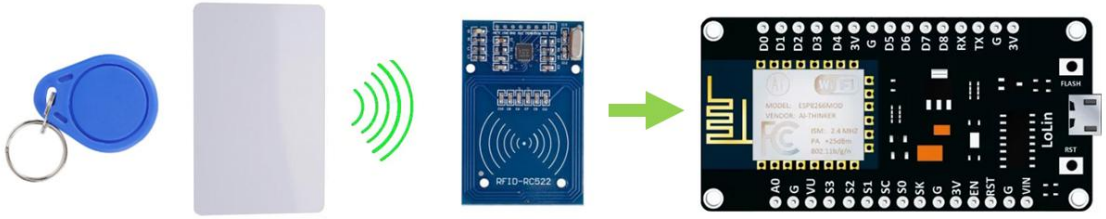


## Le tecnologie RFID

Le tecnologie **RFID** sono contrassegnate da un codice univoco, e quindi da un'identità digitale, e rappresentano una delle leve principali per lo sviluppo dell'Industria 4.0 e dell'Internet delle cose (IoT). RFID è l'acronimo di "Radio Frequency Identification" ovvero "identificazione a radiofrequenza" e si riferisce a una tecnologia in base alla quale i dati digitali codificati in un **tag RFID** vengono acquisiti da un dispostivo di lettura (**Reader)** tramite onde radio. L'RFID è simile al codice a barre ma presenta il grande vantaggio che i dati dei tag RFID possono essere letti in prossimità anche senza "contatto visivo": al contrario i codici a barre devono essere allineati con uno scanner ottico.

Un sistema RFID è costituito da due componenti principali:

1. **un transponder a radiofrequenza (Tag)** inserito nell'oggetto che si vuole identificare: il tag contiene un microchip, che memorizza ed elabora le informazioni, e un'antenna che consente la ricezione/trasmissione di dati a corto raggio senza contatto fisico. Il tag è generalmente un dispositivo passivo, ovvero non contiene una batteria
2. **un ricetrasmettitore (Reader)** controllato da un microprocessore ed usato per leggere le informazioni contenute nel tag




Tag RFID passivo Reader Sistema di gestione

- Il tag viene posizionato in prossimità del Reader (non deve necessariamente trovarsi "a vista" del Reader)
- Il Reader genera un campo elettromagnetico che fa muovere gli elettroni attraverso l'antenna del tag e alimenta il chip
- Il chip, tramite un segnale radio, trasmette al Reader le informazioni memorizzate al suo interno
- il Reader rileva e interpreta il segnale radio ricevuto e, attraverso un'interfaccia periferica seriale, invia i dati a un computer o ad un microcontrollore, nel nostro caso a ESP8266 NodeMCU, che li gestisce per realizzare gli scopi dell'applicazione (ad esempio per azionare l'apertura di una porta o di un tornello)

## Il Reader RFID RC522

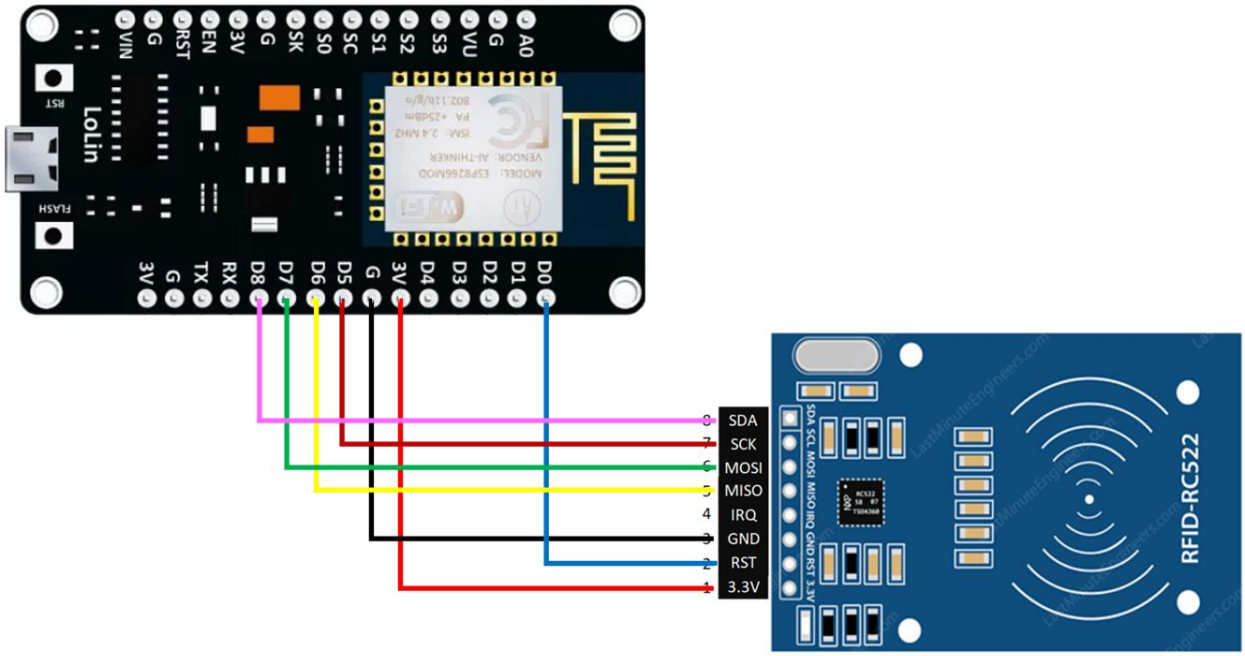
Il modulo RFID RC522 è uno dei Reader RFID più diffusi ed economici e viene fornito di solito con una o più schede RFID e uno più tag portachiavi con 1 KB di memoria. Inoltre è in grado di scrivere un tag e di memorizzarvi un testo.



L'RC522 è progettato per creare un campo elettromagnetico a 13,56 MHz, con il quale comunica con i tag RFID, e supporta la comunicazione sui protocolli SPI, I2C e UART. Nel nostro caso, comunica con ESP8266 Node MCU tramite SPI (interfaccia periferica seriale ) a 4 pin e una velocità dati massima di 10 Mb/s.
La tensione di alimentazione operativa del modulo è compresa tra 2,5 a 3,3 V , la distanza di lettura di un tag è di 5 cm.

## Piedinatura RFID RC522 e collegamenti con ESP8266 NodeMCU

| RC522 | ESP8266 NodeMCU | Colore |
|---|---|---|
| 3.3V | 3 V | |
| RST | D3 | |
| GND | GND | |
| IRQ | Non connesso | |
| MISO/ SCL / TX | D6 | |
| MOSI | D7 | |
| SCK | D5 | |
| SDA/SS/RX | D8 | |

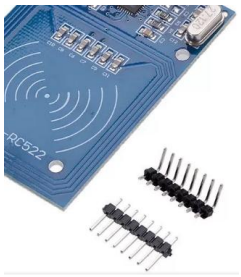


## Collegamenti su breadboard

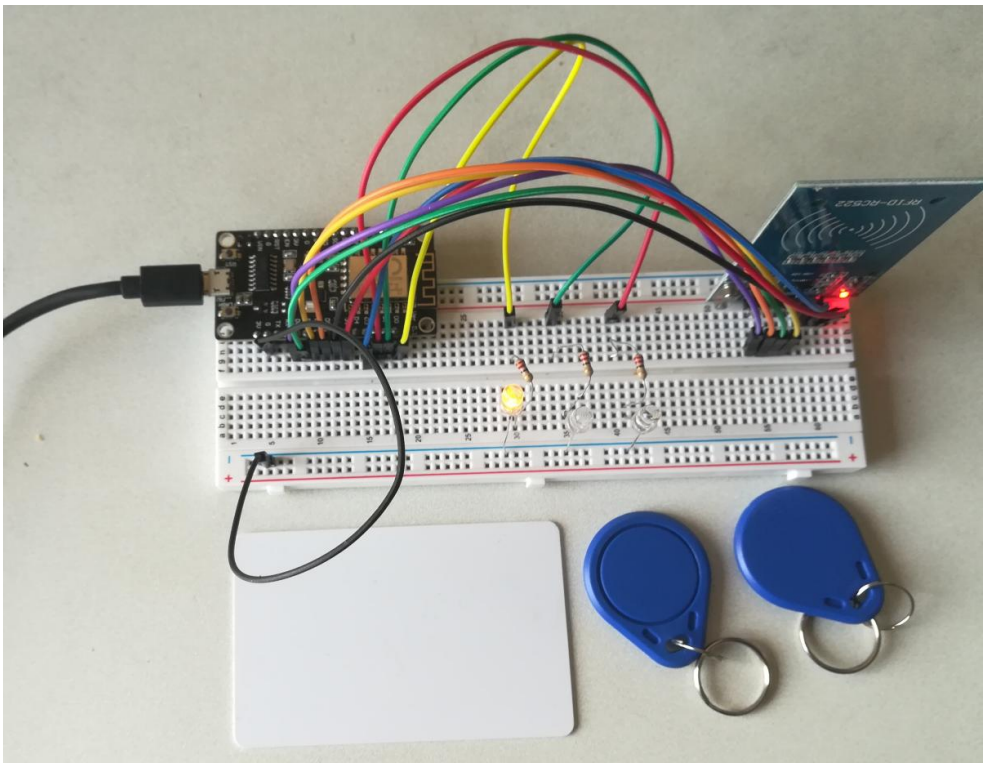
Una volta individuati i collegamenti tra ESP8266 NodeMCU e RC522, inseriamo sulla breadboard tre diodi led, ciascuno con una resistenza di limitazione di 220 Ohm:

1. il diodo led **giallo**, connesso al pin **D0**, indica che il WiFi è funzionante e il Reader è pronto. E' sempre acceso tranne quando il Reader legge il tag e per 3 secondi si accende il diodo led verde (tag riconosciuto) o il diodo led rosso (tag non riconosciuto)
2. il diodo led **verde**, connesso al pin **D1**, si accende quando il Reader legge un tag e lo riconosce
3. il diodo led **rosso**, connesso al pin **D2**, si accende quando il Reader legge un tag e non lo riconosce

Dopo aver provveduto a saldare sul dispositivo RC522 una striscia di pin (generalmente viene fornita dal Fornitore insieme con il dispositivo) ho realizzato i collegamenti tra ESP8266 NodeMCU e RC522 utilizzando una breadboard





## Componenti utilizzati

- ESP8266 NodeMCU collegato ad un PC Windows X con cavo micro USB
- Breadboard con cavetteria
- 1 lettore RFID RC522
- 2 tag portachiavi e 1 scheda RFID
- 1 diodo led giallo
- 1 diodo led verde
- 1 diodo led rosso
- 3 Resistenze da 220 Ohm



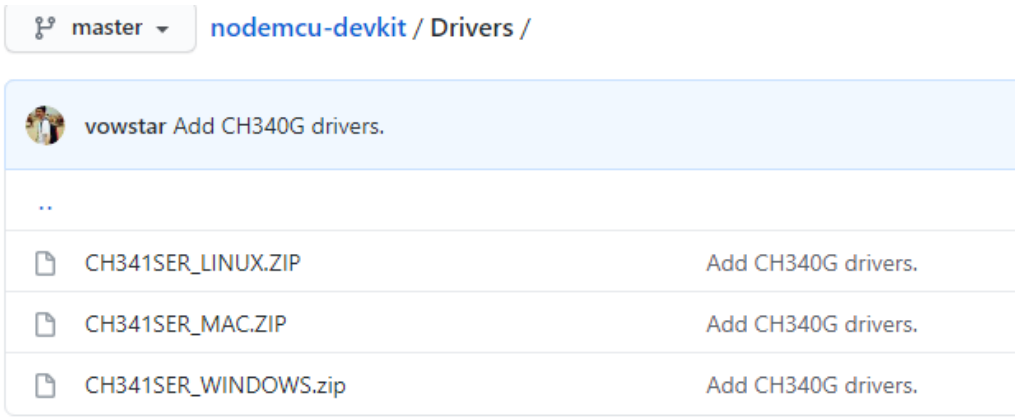
Per quel che riguarda lettore e tag, in genere per pochi euro viene fornito un kit con 3 lettori, 6 strisce di pin, 3 tag portachiavi e 3 schede RFID

## Utilizzo dell'IDE di Arduino per programmare ESP8266 NodeMCU

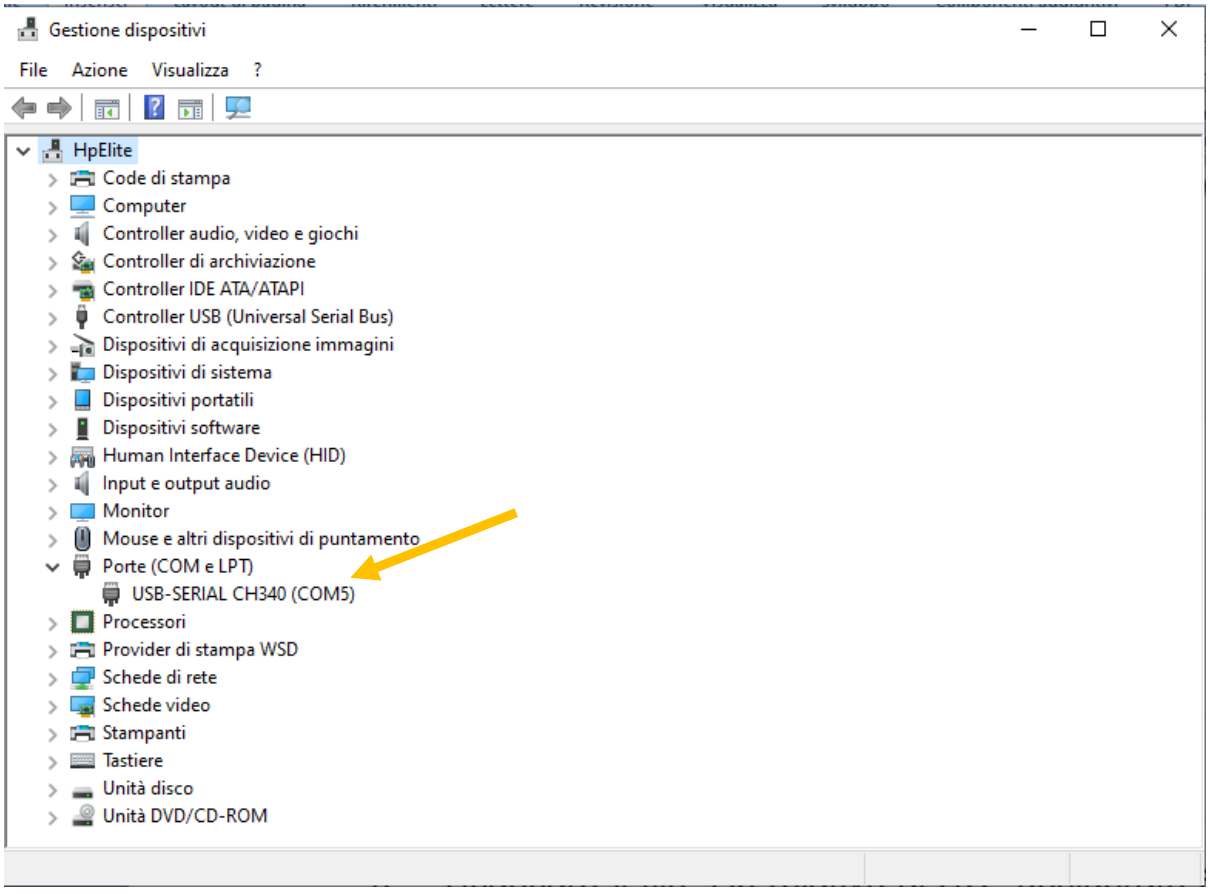
**ESP8266 NodeMCU** è un dispositivo IoT open source e per impostazione predefinita è equipaggiato con un firmware scritto con il linguaggio Lua. Spesso però, come nel nostro caso, **viene utilizzato con l'IDE di Arduino**: per questo motivo vediamo di seguito, passo passo, che cosa fare per poter programmare ESP8266 NodeMCU usando il linguaggio C ++ di Arduino.

Se già si dispone della libreria di ESP8266 andare al punto 6

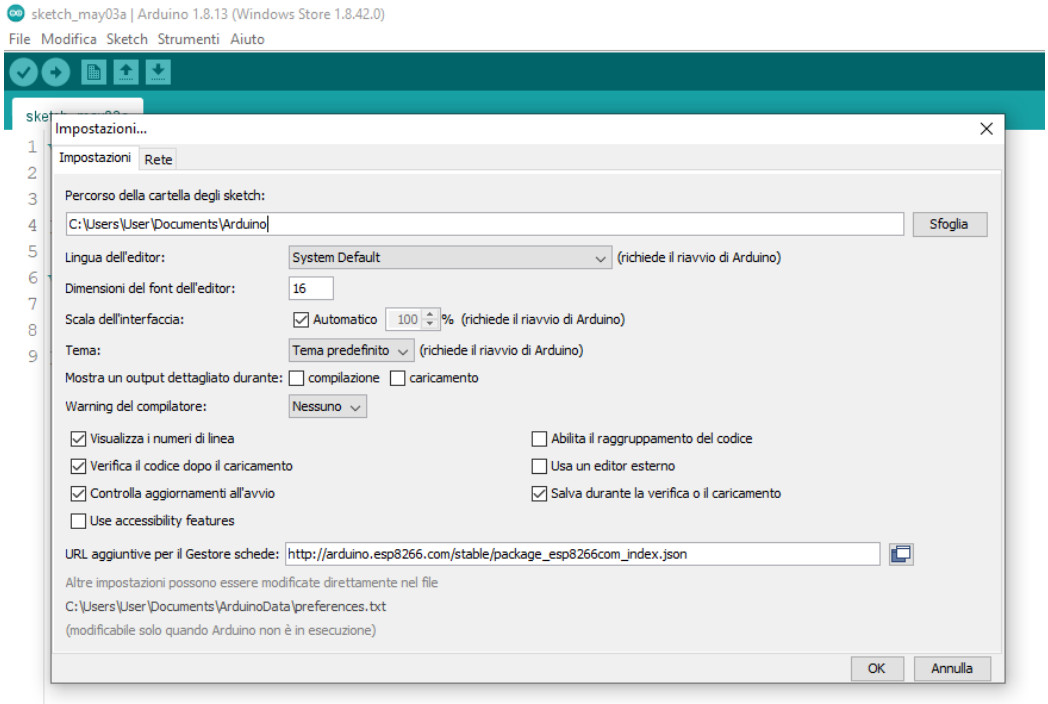
1. Collegare ESP8266 NodeMCU al PC con un cavo micro USB.
2. Aprire il link ***https://github.com/nodemcu/nodemcu-devkit/tree/master/Drivers***



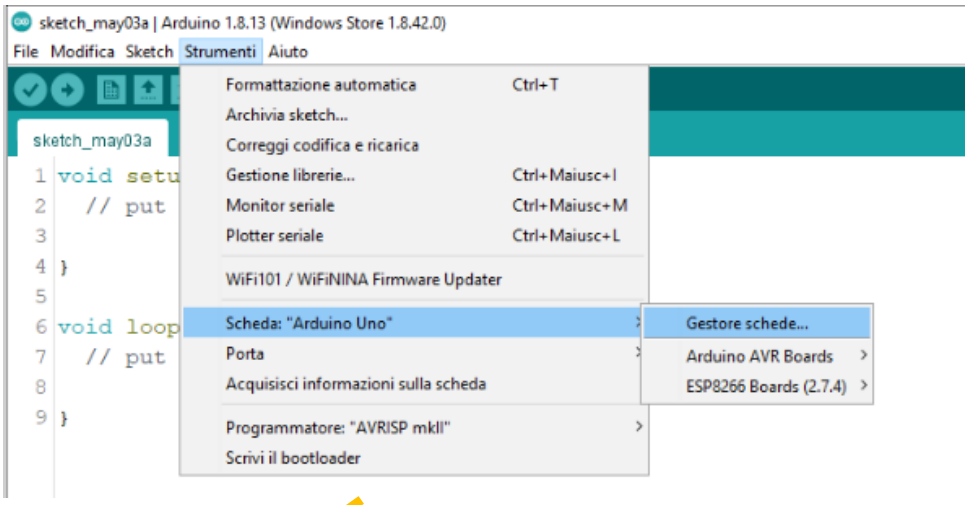

3. Scaricare il file .zip relativo al SO desiderato e scompattarlo. Eseguire quindi il file.exe che installa i driver USB per il chip USB-seriale. Se il SO è Windows, scaricare il file CH341SER_WINDOWS.zip e scompattarlo. Eseguire quindi il file CH341SER.exe e verificare l'installazione del driver, aprendo l'utility "Gestione dispositivi"

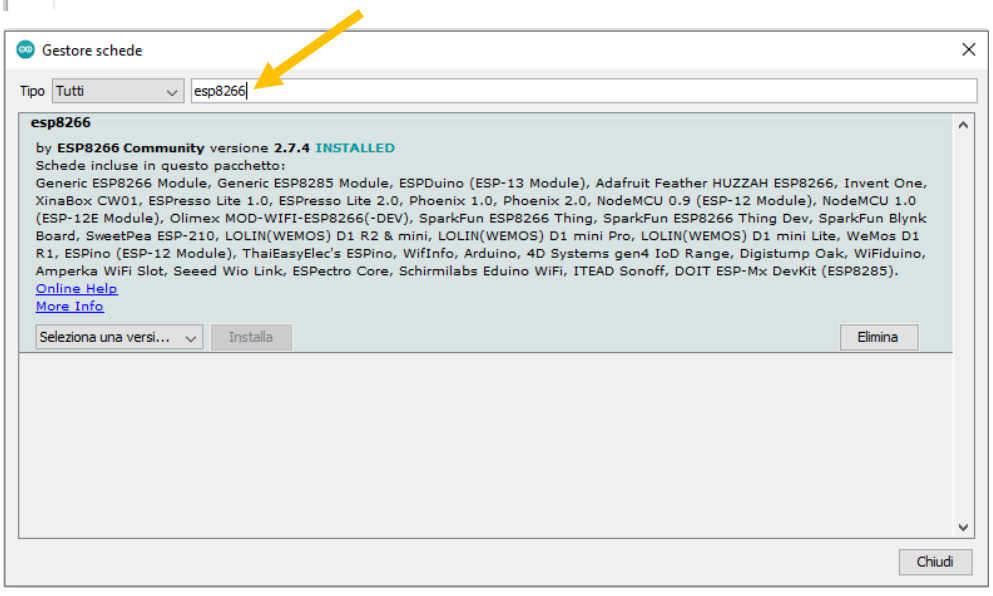


4. Aprire l'IDE Arduino, scegliere l'opzione "Impostazioni" sul menu "File" , copiare il link ***http://arduino.esp8266.com/stable/package_esp8266com_index.json*** nel campo "URL aggiuntive per il Gestore schede" e confermare cliccando su "OK"

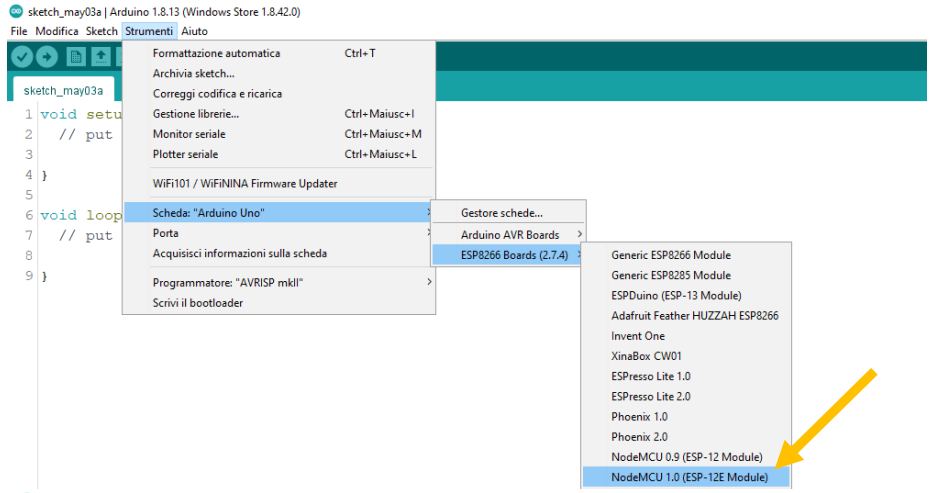


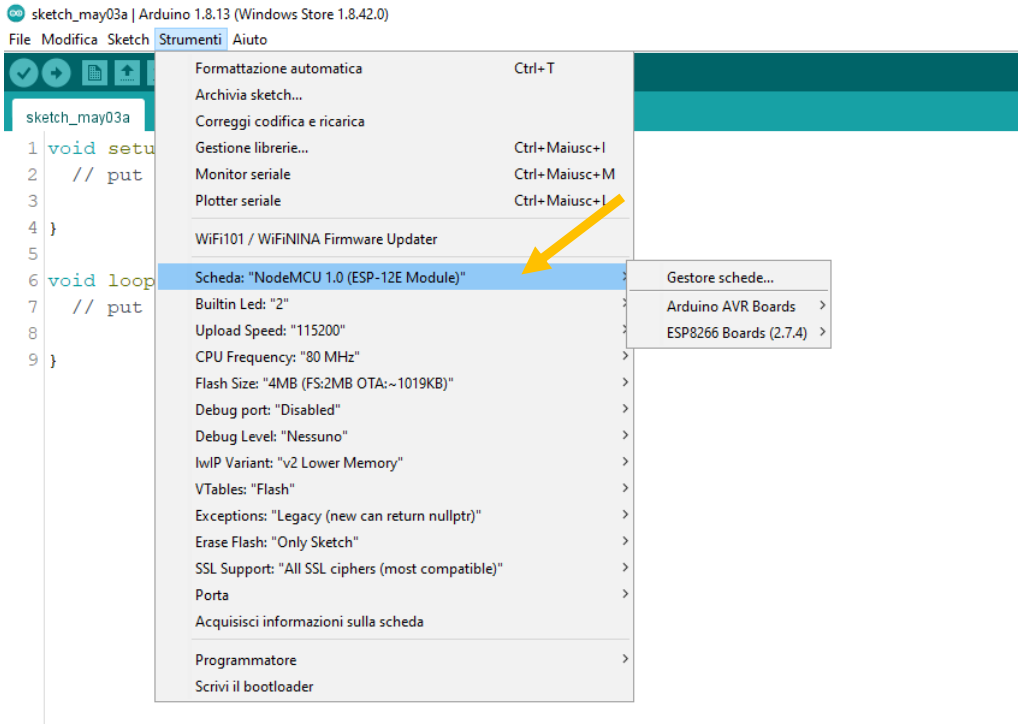

5. Sull'IDE Arduino, scegliere l'opzione "Gestore schede" del menu "Strumenti" e inserire nel campo di ricerca la parola chiave ESP8266. Installare quindi la libreria proposta.

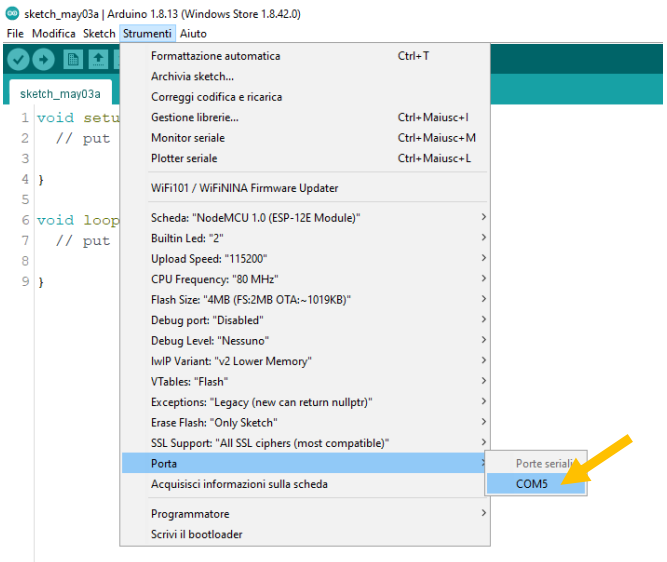


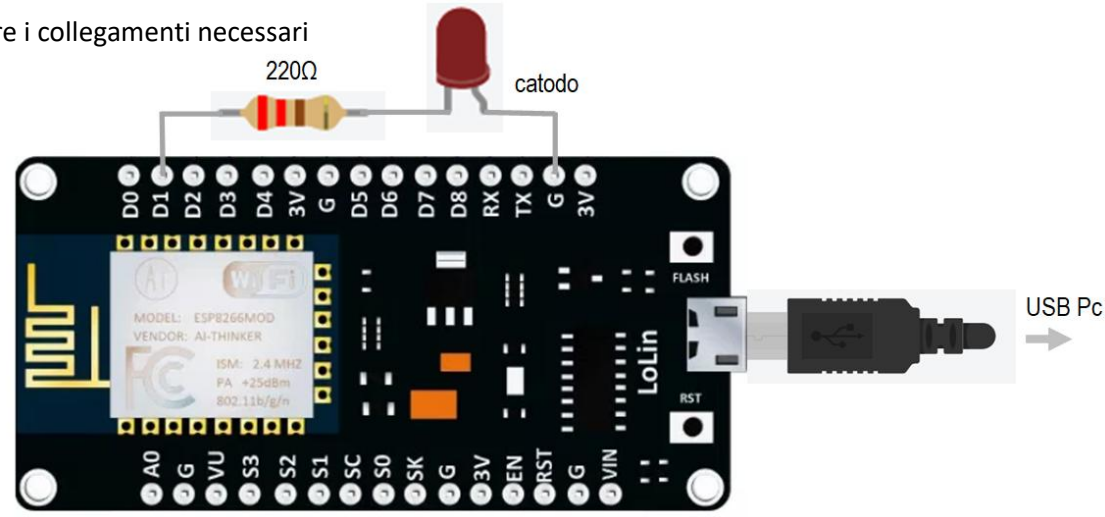
6. A questo punto, tornando all'opzione che permette di scegliere la scheda sul menu "Strumenti", vediamo che oltre alla libreria di Arduino è presente anche quella di ESP8266: la selezioniamo e scegliamo la **scheda NodeMCU (ESP -12E Module)**. Infine, sempre sul menu "Strumenti", scegliamo la porta assegnata quando sono stati installati i driver USB per il chip USB-seriale.



7. Per verificare la correttezza della configurazione dell'IDE di Arduino, proviamo ad implementare un semplice sketch che fa lampeggiare un led collegato sul pin D1
   a. Effettuare i collegamenti necessari




   b. Scrivere questo semplice sketch:

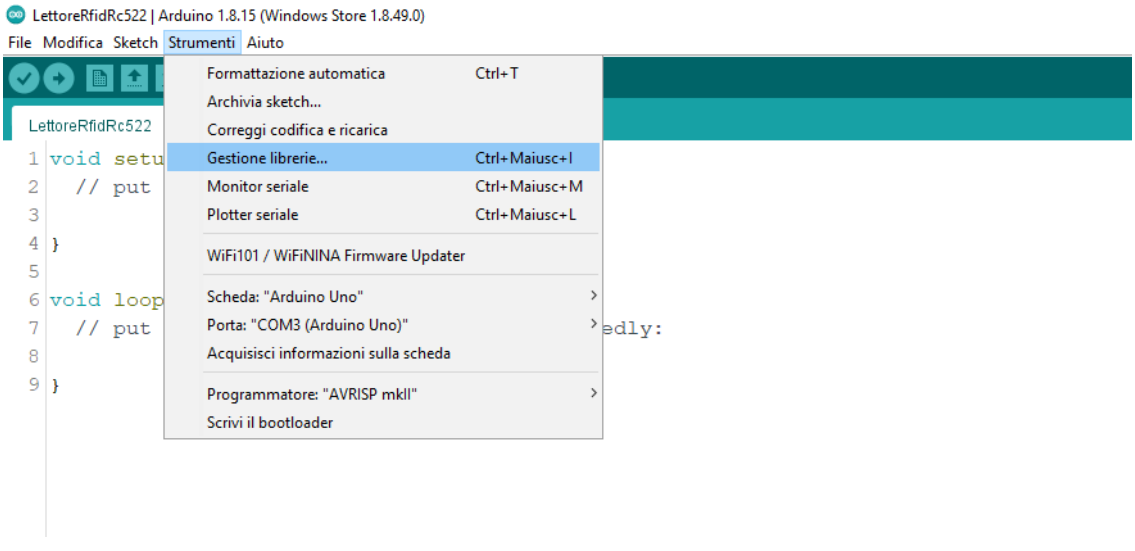
```
#define ledPin D1
void setup() {
  pinMode (ledPin, OUTPUT);
}
void loop() {
  digitalWrite(ledPin, HIGH);
  delay(1000);
  digitalWrite(ledPin, LOW);
  delay(1000);
}
```

   c. Collegare l'ESP8266 NodeMCU al computer utilizzando il cavo micro USB. Selezionare la scheda NodeMCU (ESP -12E Module) e la porta assegnata
   d. Caricare il programma: il diodo led collegato al pin D1 inizia a lampeggiare

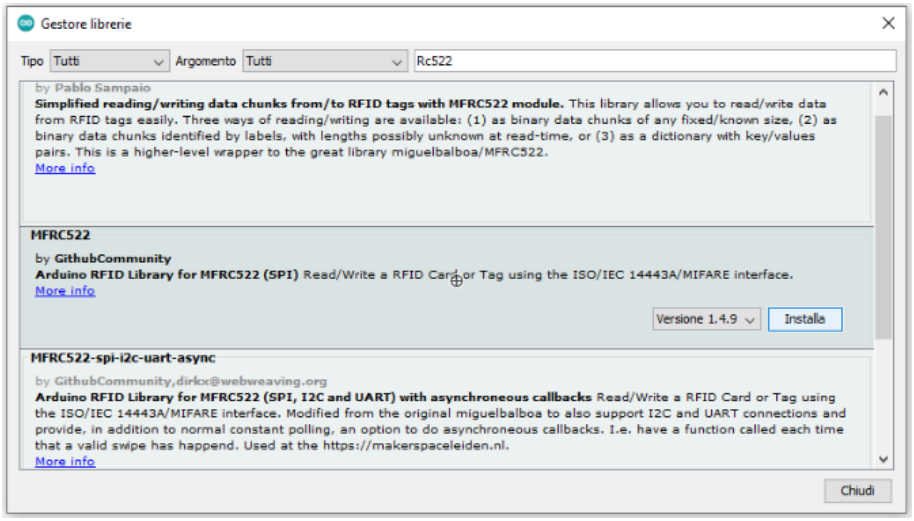
## La Libreria MFRC522

Prima di iniziare la programmazione con l'IDE di Arduino, e se non si dispone già della libreria MFRC522, bisogna installare, e quindi includere nello sketch, la libreria necessaria per poter utilizzare il lettore di tag RFID RC522.

*1.* Con l'IDE Arduino in esecuzione, scegliere *Strumenti-Gestione librerie…*



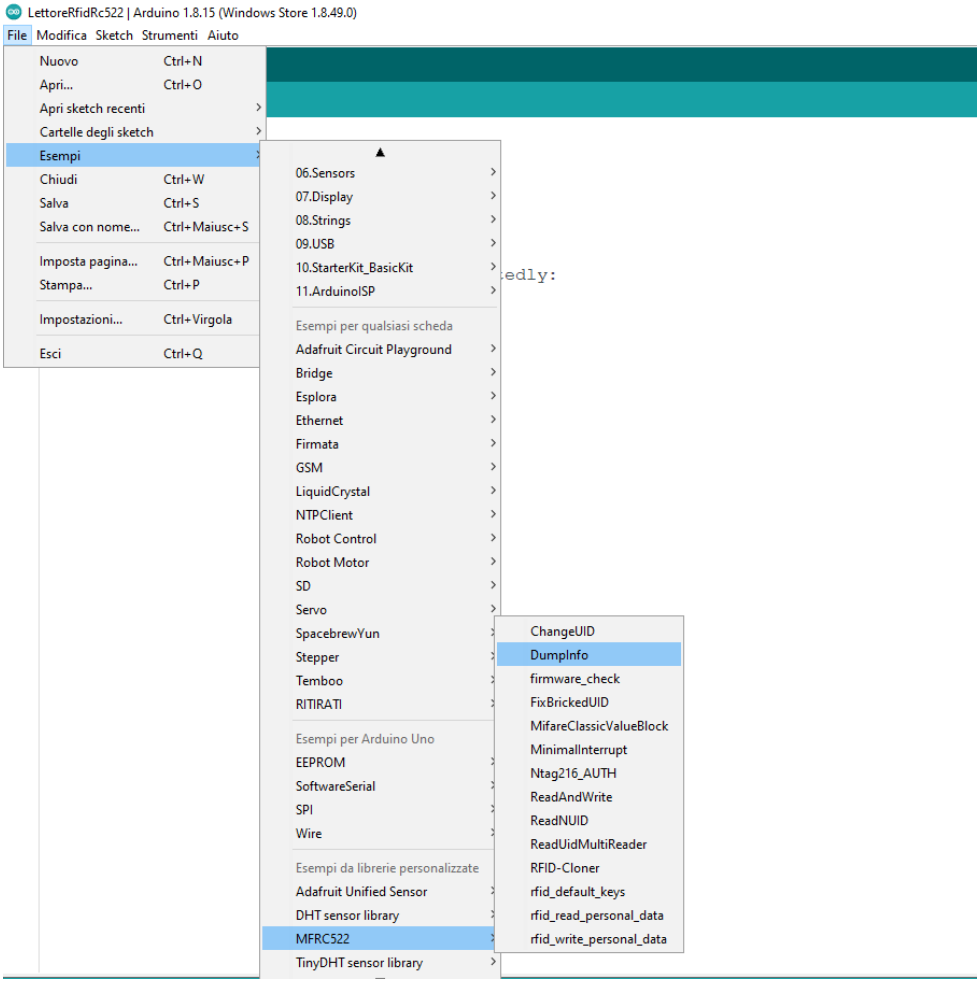
2. Quando si apre la finestra del Gestore librerie, nel campo di ricerca inserire "RC522"
3. Tra le diverse librerie proposte trovare, selezionare e installare "MFRC522" ( by GithubCommunity)




## La lettura dei codici identificativi dei tag a nostra disposizione

Una volta installata la libreria MFRC522, utilizziamo uno sketch della libreria che ci permette di leggere il contenuto dei tag a nostra disposizione, in particolare ci permette di conoscere il codice identificativo del tag. Questi codici saranno inseriti nel database MySql remoto che sarà consultato da ESP8266 Node MCU e ci permetteranno di riconoscere o meno i tag letti dal Reader RC522

Dall'opzione **File->Esempi** selezionamo **MFRC522->DumpInfo**



Questo sketch, come suggerisce il suo nome, legge il tag e ci fornisce i dati in esso memorizzati, in particolare il codice identificativo. Lo sketch deve essere modificato (come si vedrà in seguito) in quanto è progettato per Arduino Uno e non per ESP8266 NodeMCU (in particolare sono diversi i collegamenti con RC522)

Avvicinando il tag al modulo (facendo attenzione a non spostarlo fin quando l'operazione di lettura non è completata), otteniamo sul monitor seriale tutti i dati in esso memorizzati. Facciamo questa operazione per uno dei due tag portachiavi utilizzati e per una scheda RFID, che nel nostro progetto saranno riconosciuti dal sistema (accensione del led verde). L'altro tag portachiavi servirà ad illustrare il caso in cui il sistema non riconosce il tag e non autorizza alcuna azione (accensione del led rosso)

## Codice dello sketch DumpInfo

**Attenzione**: **ho modificato lo sketch DumpInfo sostituendo**

la riga

Serial.begin(9600)

con la riga

Serial.begin(115200)

e le righe

#define RST_PIN 9
#define SS_PIN 10

con le righe

#define RST_PIN D3
#define SS_PIN D8

```
#include <SPI.h>
#include <MFRC522.h>

#define RST_PIN         D3          // Configurable, see typical pin layout above
#define SS_PIN          D8          // Configurable, see typical pin layout above

MFRC522 mfrc522(SS_PIN, RST_PIN);  // Create MFRC522 instance

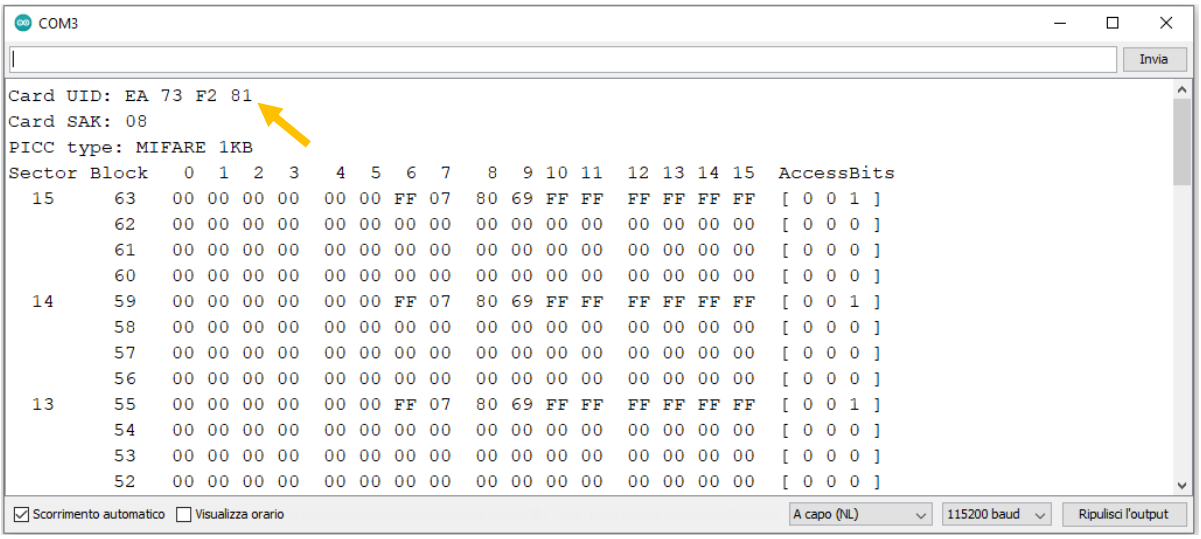
void setup() {
            Serial.begin(115200);               // Initialize serial communications with the PC
            while (!Serial);                    // Do nothing if no serial port is opened (added for Arduinos based on
ATMEGA32U4)
            SPI.begin();                                // Init SPI bus
            mfrc522.PCD_Init();                 // Init MFRC522
            delay(4);                                   // Optional delay. Some board do need more time after init to be
ready, see Readme
            mfrc522.PCD_DumpVersionToSerial(); // Show details of PCD - MFRC522 Card Reader details
            Serial.println(F("Scan PICC to see UID, SAK, type, and data blocks..."));
}

void loop() {
            // Reset the loop if no new card present on the sensor/reader. This saves the entire process when idle.
            if ( ! mfrc522.PICC_IsNewCardPresent()) {
                        return;
            }

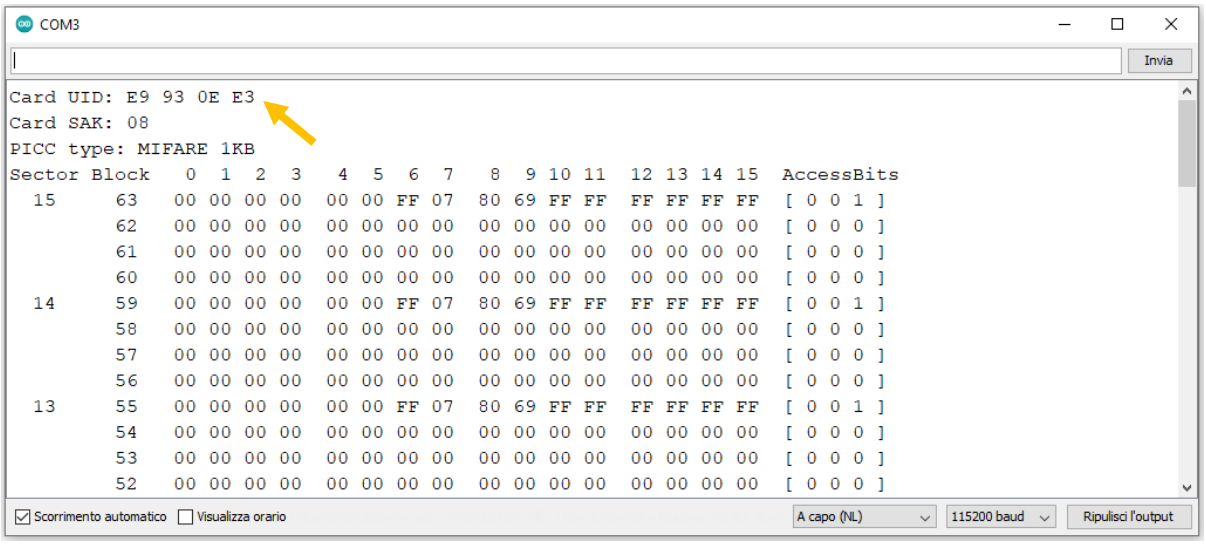
            // Select one of the cards
            if ( ! mfrc522.PICC_ReadCardSerial()) {
                        return;
            }

            // Dump debug info about the card; PICC_HaltA() is automatically called
            mfrc522.PICC_DumpToSerial(&(mfrc522.uid));
}
```

Avviamo lo sketch DumpInfo, apriamo il monitor seriale, impostiamo la velocità a 115200 baud e avviciniamo il portachiavi RFID. Otteniamo il dump dei dati da cui ricaviamo il **Card UID**, ovvero il codice identificativo esadecimale, EA73F281 (8 cifre esadecimali, ovvero 4 byte)




Avviciniamo la scheda RFID. Otteniamo il dump dei dati da cui ricaviamo il **Card UID**, ovvero il codice identificativo esadecimale, E9930EE3 (8 cifre esadecimali, ovvero 4 byte)

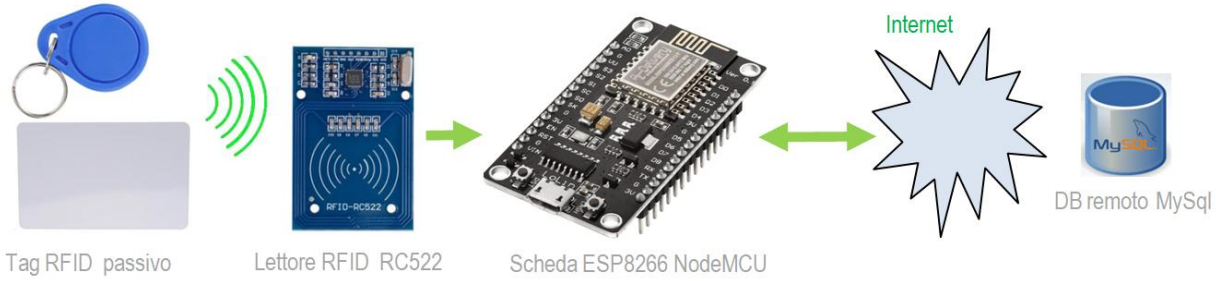



Evitiamo di descrivere in questo progetto il significato e le modalità di interpretazione di tutti i dati che compaiono nel monitor seriale. Mi preme invece consigliare la buona pratica di consultare sempre gli esempi delle librerie utilizzate con l'IDE di Arduino: si tratta di ottimi programmi che spiegano le principali funzionalità delle librerie e spesso rappresentano un'eccellente base di partenza per le applicazioni che vogliamo realizzare.

## Schema del progetto

Vogliamo realizzare un sistema basato sulla scheda ESP8266 NodeMCU interfacciata con un Reader RC522. Il sistema:

1. legge un tag e effettua una richiesta HTTP GET a uno script PHP che si connette al DataBase MySql 'rfid', verifica se il codice letto è valido, ovvero se è presente nella tabella 'codicirfid', e memorizza la lettura nella tabella 'letturerfid'
2. se il codice è valido accende un diodo led verde, viceversa accende un diodo led rosso
3. visualizza le letture su una pagina web




Per la realizzazione e il test del progetto ho utilizzato:

- un DataBase MySql *'rfid'* sul Server Linux che ospita il mio sito web www.maurodeberardis.it con due tabelle *'codicirfid'* e *'letturerfid'*
- L'applicazione Arduino su ESP8266 NodeMCU *esp8266_rfid.ino*
- lo script PHP lato server *www.maurodeberardis.it/rfid/validarfid.php*
- la pagina web *www.maurodeberardis.it/rfid/displayletturerfid.php*

**In alternativa al Server di hosting remoto si può utilizzare un Server locale**. Per questo motivo ho testato il progetto anche con un server locale WAMP e nei codici di programmazione, che fanno riferimento al test con Server remoto *www.maurodeberardis.it*, ho inserito anche le righe commentate da utilizzare con WAMP (in questo caso, l'indirizzo del pc che ospita Wamp Server è 192.168.1.3)

## Query di creazione del DB MySql *'rfid'* e delle tabelle *'codicirfid'* e *'letturerfid'*

Con phpMyAdmin o un altro software, sul Server di hosting remoto o su Wamp Server, eseguiamo le query di creazione di DataBase e tabelle necessarie per il progetto.

```
CREATE DATABASE rfid;
```

```
CREATE TABLE IF NOT EXISTS codicirfid (
id INT(10) PRIMARY KEY AUTO_INCREMENT,
codicetag char(8))
```

```
CREATE TABLE IF NOT EXISTS letturerfid (
id INT(10) PRIMARY KEY AUTO_INCREMENT,
codicetag char(8),
dataora datetime,
riconosciuto varchar(1))
```

Nella tabella *'codicirfid'* inserisco i codici identificativi esadecimali dei due tag "riconosciuti" con DumpInfo: EA73F281 del tag portachiavi e E9930EE3 della Scheda RFID

```
INSERT INTO codicirfid (codicetag)
VALUES ("EA73F281"),("E9930EE3")
```

| | | | | id | codicetag |
|---|---|---|---|---|---|
| ☐ | Modifica | Copia | Elimina | 1 | EA73F281 |
| ☐ | Modifica | Copia | Elimina | 2 | E9930EE3 |

# L'applicazione Arduino su ESP8266 NodeMCU 'esp8266_rfid.ino'

```
/*
 * ----------------------------------------------------
 * Prof. Mauro De Berardis 2021
 * ----------------------------------------------------
 * Utilizzando la scheda ESP8266 NodeMCU e l'IDE di Arduino, realizziamo un progetto che  permette
 * di leggere un tag RFID e "riconoscerlo" consultando un DataBase MySql  su un server remoto
 * ESP8266  è interfacciato  con un  dispositivo RC522  che  legge un tag RFID (una carta magnetica o
 * un tag portachiavi). Il codice letto viene inviato ad un DB MySql remoto: se è presente nel database,
 * si  accende un diodo led verde, viceversa si accende un diodo led rosso.
 * Una pagina web, che si può visualizzare accedendo al server  da qualsiasi luogo e con qualsiasi
 * dispositivo, mostra le letture effettuate dal Reader.
 * Al posto della carta magnetica  e del tag portachiavi si può utilizzare un qualsiasi oggetto con
 * tag RFID.
 * L'accensione del diodo verde simula l'azione che viene intrapresa: l'azionamento di un relé per aprire
 * una porta o per accendere un motore, il tracciamento dell'oggetto, l'aggiornamento del magazzino,
 * il monitoraggio dell'avanzamento della produzione, ecc.
 *
 * -----------------------------------
 *             MFRC522      ESP8266
 *             Reader/PCD   NodeMCU
 * Segnale     Pin          Pin
 * -----------------------------------
 * RST/Reset   RST          D3
 * SPI SS      SDA(SS)      D8
 * SPI MOSI    MOSI         D7
 * SPI MISO    MISO         D6
 * SPI SCK     SCK          D5
 */

//***Include  librerie  per ESP8266 NodeMCU
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266HTTPClient.h>

//***Include librerie per RFID
#include <SPI.h>
#include <MFRC522.h>

#define Led_Giallo D0
#define Led_Verde  D1
#define Led_Rosso  D2
#define RST_PIN    D3 // pin di ESP8266 NodeMcu connesso con pin RST di RC522
#define SS_PIN     D8 // pin di ESP8266 NodeMcu connesso con pin SS/SDA/RX  di RC522

//***Configurazione WiFI: impostazione delle credenziali di rete
const char* ssid = "mySSID";         //Inserire qui l'SSID
const char* password = "mYPassword"; //Inserire qui la password WiFi

// Crea un'istanza della classe MFRC522 che chiamiamo mfrc522
MFRC522 mfrc522(SS_PIN, RST_PIN);

String tagLetto="";
bool lettura=true;
void setup()
{
  // apre una connessione seriale. A scopo di debug inviamo messaggi al monitor seriale
  Serial.begin(115200);
  SPI.begin(); //  inizializza SPI bus
  mfrc522.PCD_Init(); // inizializza il lettore  RC522
  pinMode(Led_Giallo,OUTPUT);        // led giallo
  pinMode(Led_Verde,OUTPUT);         // led verde
  pinMode(Led_Rosso,OUTPUT);         // led rosso

  WiFi.begin(ssid, password);   //Si connette al router WiFi
  Serial.println("Connessione in corso");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("ESP8266 NodeMCU connessa alla rete WiFI: ");
  Serial.print(ssid);
  Serial.print("  con il numero IP : ");
  Serial.println(WiFi.localIP());//l'IP di ESP8266 viene assegnato automaticamente da DHCP
  Serial.println();

// led giallo acceso indica che il WiFi è attivo e il Reader è in attesa di leggere un tag
  digitalWrite(Led_Giallo, HIGH);
  delay(1000);
}
```

```
void loop()
{
   while (getUID())  //Rimane in attesa di letture da parte del Reader RC522
  {
    if (tagLetto!="")
    {
      HTTPClient http;    //Dichiara l'oggetto http della classe HTTPClient
      String  richiestaHTTP, link,risposta;

      richiestaHTTP = "?codice=" + tagLetto ; //querystring da inviare nell'URL della richiesta HTTP GET

     //se utilizzo un server di hosting remoto ad esempio www.maurodeberardis.it
     link= "http://www.maurodeberardis.it/rfid/validarfid.php"+richiestaHTTP;

     //se utilizzo Wamp Server
     //link = "http://192.168.1.3/rfid/validarfid.php"+richiestaHTTP ;

      http.begin(link);           //Specifica la destinazione della richiesta HTTP GET

      int httpCode = http.GET();  //Invia la richiesta HTTP GET

      if (httpCode==HTTP_CODE_OK)
      {

          risposta= http.getString(); //Ottiene la risposta dal Server
          Serial.print("HTTP code: ");
          Serial.println(httpCode);
          Serial.print("Codice tag letto: ");
          Serial.println(tagLetto);
          // in base al primo carattere della risposta del Server (S/N, riconosciuto/non riconosciuto)
          // attiva una determinata azione (led verde/rosso, apertura/non apertura ecc.)
          azione(risposta);
      }
      else
      {
          Serial.println("Errore");
          Serial.print("HTTP code: ");
          Serial.println(httpCode);
          Serial.println(link);
          Serial.println();
      }
    }//chiude if tagLetto------
  }//chiude while -----------------
}//chiude loop() ----------------------

//Lettura
boolean getUID()
{
  tagLetto = "";

  if(!lettura) return false;  // il Reader non può fare letture per 3s da led verde o led rosso ON
  if ( ! mfrc522.PICC_IsNewCardPresent()) return false; // verifica se viene rilevato un tag RFID
  if ( ! mfrc522.PICC_ReadCardSerial())   return false; // verifica se i dati possono essere letti dal tag
RFID

  // legge i dati del tag
  for ( byte i = 0; i < 4; i++) // il Card UID contiene 4 byte ovvero 8 num esadecimali(ad es. E9 93 0E E3)
  {
    int n=mfrc522.uid.uidByte[i]; // ottiene il numero intero n corrispondente a ciascun byte del Card UID
    String s=String(n, HEX);      // converte n in una stringa s di numeri esadecimali
    if(s.length()<2)s="0"+s;      // se la stringa s contiene un solo carattere aggiunge lo 0 iniziale
    tagLetto.concat(s);           // concatena la stringa s al tagLetto;
  }
  tagLetto.toUpperCase();         // converte la stringa tagLetto in maiuscole
  mfrc522.PICC_HaltA();           // termina la lettura
  return true;
}

void azione(String s)
{
  String riconosciuto=s.substring(0,1);
  String dataora=s.substring(1);
  if (riconosciuto=="S")
  {
     Serial.println("Tag  riconosciuto");
     Serial.print("Card UID letto: ");
     Serial.println(tagLetto);
     Serial.print("Data e ora lettura: ");
     Serial.println(dataora);
     Serial.println();
     // inibisce la lettura  e spegne il Led_Giallo e accende il Led_Verde per 3 secondi
     lettura=false;
     digitalWrite(Led_Giallo, LOW);
     digitalWrite(Led_Verde, HIGH);
```

```
    delay(3000);
    digitalWrite(Led_Verde, LOW);
    lettura=true; //ri-abilita la lettura e ri-accende il Led_Giallo
    digitalWrite(Led_Giallo, HIGH);
   }
   else
   {
    Serial.println("------------------------ Tag non riconosciuto");
    Serial.print(  "------------------------ Card UID letto: ");
    Serial.println(tagLetto);
    Serial.print(  "------------------------ Data e ora lettura: ");
    Serial.println(dataora);
    Serial.println();
    // inibisce la lettura e spegne il Led_Giallo e  accende il Led_Rosso per 3 secondi e
    lettura=false;
    digitalWrite(Led_Giallo, LOW);
    digitalWrite(Led_Rosso, HIGH);
    delay(3000);
    digitalWrite(Led_Rosso, LOW);
    lettura=true; //ri-abilita la lettura e ri-accende il Led_Giallo
    digitalWrite(Led_Giallo, HIGH);
  }
}
```

## Gli script lato server 'validarfid.php' e 'displayletturerfid.php'

### Script 'validarfid.php'

1. recupera il codice del tag inviato con una richiesta HTTP GET da ESP8266 NodeMCU lato client
2. esegue una query di ricerca per verificare se il codice è presente nella tabella *'codicirfid'*: se è presente memorizza la lettura nella tabella *'letturerfid'* e restituisce il valore 'S' , se non è presente restituisce il valore 'N'. Restituisce inoltre la data e l'ora della lettura.

```
<?php

$conn=mysqli_connect("myhost","myusername","mypassword","mydatabase");
        /* $conn=mysqli_connect("192.18.1.3","myusername","mypassword","rfid");
        se utilizzo un Server locale*/
        //      192.18.1.3 è l'indirizzo IP del Server Wamp utilizzato*/

$ora = date("Y-m-d H:i:s");

if(!empty($_GET['codice']))
 {
        $codiceletto=$_GET['codice'];
    $s="Select * from codicirfid where codicetag='$codiceletto'";
    $q=mysqli_query($conn,$s);
    if(mysqli_num_rows($q) >0)
        {
                //tag riconosciuto
                $riconosciuto="S";
        }
        else
        {
                $riconosciuto="N";
        }

    $s="Insert into letturerfid (codicetag,dataora,riconosciuto)
        values ('$codiceletto','$ora','$riconosciuto')";
    $q=mysqli_query($conn,$s);
    echo $riconosciuto.$ora;
}
mysqli_close($conn);

?>
```

## Script 'displayletturerfid.php'

La pagina www.maurodeberardis.it/rfid/displayletturerfid.php (o 192.168.1.3/rfid/displayletturerfid.php) visualizza i dati memorizzati nella tabella "letturerfid" del database "rfid". La pagina, molto semplice ed essenziale, stampa in una tabella le letture, eventualmente filtrate per data, a partire dalla lettura più recente.

```
<!DOCTYPE html>
<html>
<head><title>ESP8266-RC522-MySql </title></head>
<style>
 td,th{border:1px solid #aaaaaa;text-align:center}
</style>
<script language="JavaScript">
function valida()
{
        dl=document.form1.data_let.value;
        if(dl=="")
        {
                alert("Inserire la data");
                return false;
        }
}
</script>
<body>
<?php
if(!isset($_GET["submit"]))
{  ?>
   <form name="form1" onsubmit="return valida()" >
        <h2>Letture tag RFID</h2>
        <label>Data (data vuota: tutte le letture)</label><br/>
        <input type="date" id="data_let" name="data_let">
        <br/><br/>
        <input type="submit" name="submit" value="Display dati">
        <input type="reset" value="Reset">
    </form>
<?php
}
else
{
        $data_let= $_GET['data_let'];
        $aa=substr($data_let,0,4);
        $mm=substr($data_let,5,2);
        $gg=substr($data_let,8,2);
        $data=$gg."/".$mm."/".$aa;
        $aa=(int)$aa;    $mm=(int)$mm;    $gg=(int)$gg;
        $titolo="Data lettura RFID: ".$data ;
        if($aa=="")$titolo="Letture RFID";

        $conn=mysqli_connect("myhost","myusername","mypassword","mydatabase");
        /* $conn=mysqli_connect("192.18.1.3","myusername","mypassword","rfid");
        se utilizzo un Server locale*/
        //      192.18.1.3 è l'indirizzo IP del Server Wamp utilizzato*/


        if($aa=="")
        {
                $s="Select * from letturerfid order by dataora Desc";
        }
        else
        {
                $s="Select * from letturerfid where
                Year(dataora)=$aa And Month(dataora)=$mm And Day(dataora)=$gg
                order by dataora Desc";
        }

        $q=mysqli_query($conn,$s);
```

```
        $nr=mysqli_num_rows($q);

        if ($nr==0)
        {
                echo "<br><br>Nessun dato trovato<br/><br/>";
        }
        else
        {
                echo "<h1>$titolo</h1>";
                echo "<table width=80%>";
                echo "<tr><th>Nr</th>
                        <th>Id</th>
                        <th>Time lettura</th>
                        <th>Codice Tag</th>
                        <th>Valido</th>
                      </tr>";
                $k=0;
                while($r=mysqli_fetch_array($q))
                {
                        $k++;
                        $timemisura=substr($r[2],11,8);
                        if($aa=="")$timemisura=$r[2];
                        echo "<tr><td>$k</td><td>$r[0]</td>
                        <td>$timemisura</td>
                        <td>$r[1]</td>";
                        if($r[3]=="S")
                        {
                                echo"<td>$r[3]</td>";
                        }
                        else
                        {
                                echo"<td style='background:#d3d3d3'>$r[3]</td>";
                        }
                        echo"</tr>";
                }
                echo"</table>";
        }
        mysqli_close($conn);
        echo"<br/><br/><a href='displayletturerfid.php'>Scegli un'altra data</a>";
}
?>
</body>
</html>
```

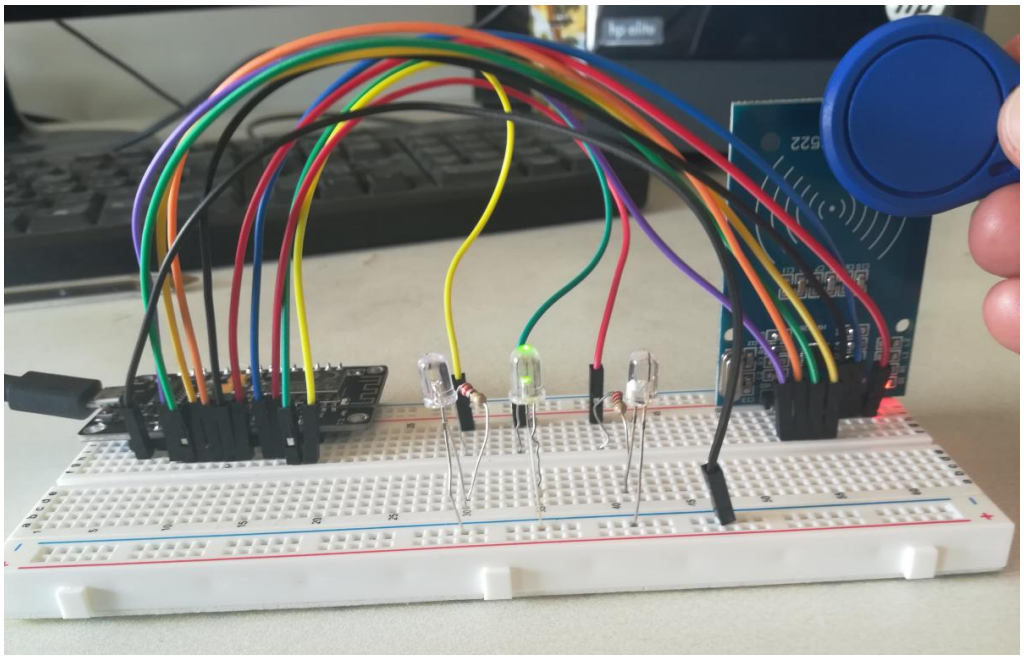
## Scarica la cartella rar con i tre codici del progetto

# Prova del progetto

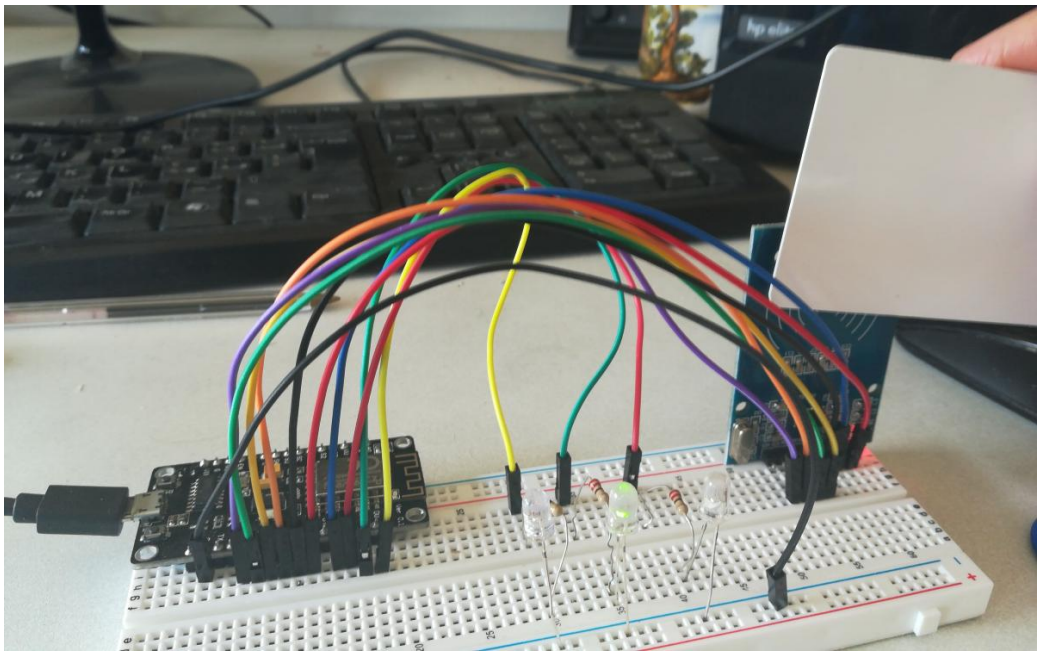
## Letture RFID

Il progetto è perfettamente funzionante sia con server di hosting remoto (www.maurodeberardis.it) sia con Wamp Server (192.168.1.3): qui vengono riportate le **prove fatte con il server di hosting remoto**.

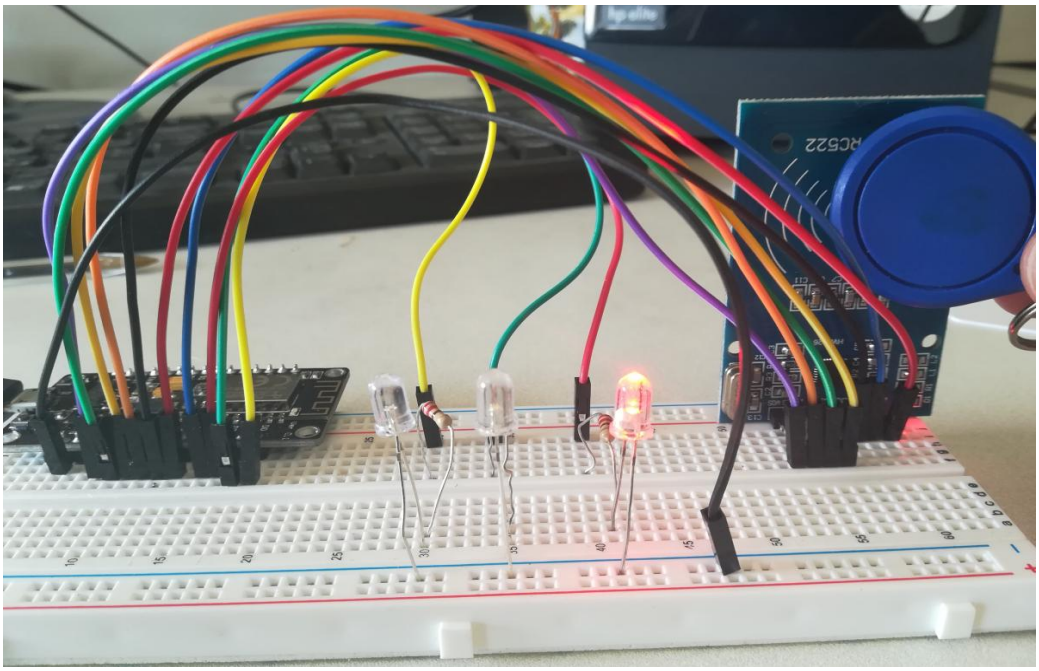
Resetto la scheda ESP8266 NodeMCU (premendo l'apposito pulsantino) e sull'IDE Arduino apro il monitor seriale (la velocità deve essere impostata a 115200 baud): il diodo led giallo indica che la scheda Esp8266 NodeMCU è connessa alla rete WiFi e che il Reader RFID RC522 è in attesa di rilevare un tag.
Avvicino al Reader il tag portachiavi precedentemente identificato con DumpInfo: **il tag viene riconosciuto**, il diodo led giallo si spegne e il diodo led **verde** si accende per 3 secondi



Avvicino al Reader la scheda RFID precedentemente identificata con DumpInfo: **il tag viene riconosciuto**, il diodo led giallo si spegne e il diodo led **verde** si accende per 3 secondi



Avvicino al Reader l'altro tag portachiavi non identificato con DumpInfo: **il tag non viene riconosciuto**, il diodo led giallo si spegne e il diodo led **rosso** si accende per 3 secondi



Le tre letture sono riportate sul monitor seriale e memorizzate nella tabella *'letturerfid'* del DB MySql remoto:

```
.
ESP8266 NodeMCU connessa alla rete WiFI: TIM-24478936  con il numero IP : 192.168.1.123

HTTP code: 200
Codice tag letto: EA73F281
Tag  riconosciuto
Card UID letto: EA73F281
Data e ora lettura: 2021-09-25 11:28:12

HTTP code: 200
Codice tag letto: E9930EE3
Tag  riconosciuto
Card UID letto: E9930EE3
Data e ora lettura: 2021-09-25 11:28:31

HTTP code: 200
Codice tag letto: EA72FD80
------------------------ Tag non riconosciuto
------------------------ Card UID letto: EA72FD80
------------------------ Data e ora lettura: 2021-09-25 11:28:45
```

IP assegnato automaticamente da DHCP

Data e ora lettura memorizzate su DB MySql

Scorrimento automatico | Visualizza orario | A capo (NL) | 115200 baud | Ripulisci l'output

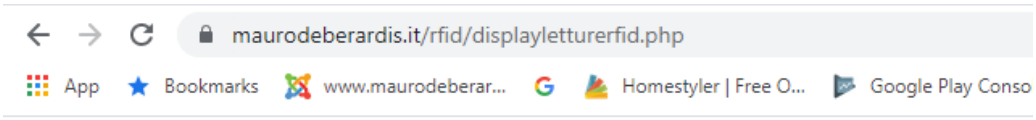
| | | id | codicetag | dataora | riconosciuto |
|---|---|---|---|---|---|
| ☐ | Modifica Copia Elimina | 23 | EA73F281 | 2021-09-25 11:28:12 | S |
| ☐ | Modifica Copia Elimina | 24 | E9930EE3 | 2021-09-25 11:28:31 | S |
| ☐ | Modifica Copia Elimina | 25 | EA72FD80 | 2021-09-25 11:28:45 | N |

Seleziona tutti / Deseleziona tutti *Se selezionati:* Modifica Elimina Esporta

**Mostra :** Start row: 0 Number of rows: 30 Headers every 100 rows

## Display letture

Sul PC desktop

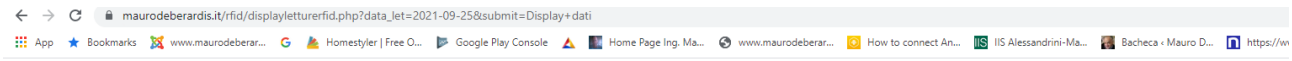



# Letture tag RFID

Data (data vuota: tutte le letture)

25/09/2021

Display dati Reset




# Data lettura RFID: 25/09/2021

| Nr | Id | Time lettura | Codice Tag | Valido |
|---|---|---|---|---|
| 1 | 25 | 11:28:45 | EA72FD80 | N |
| 2 | 24 | 11:28:31 | E9930EE3 | S |
| 3 | 23 | 11:28:12 | EA73F281 | S |

Scegli un'altra data

Sullo smartphone

