

Traintorio

Informe técnico

Un trabajo presentado para la materia de
Proyectos y Diseño Electrónico



Krapp Ramiro

Instituto tecnológico San Bonifacio
Departamento de electrónica
27 de marzo de 2022

Hecho en L^AT_EX
Versión Alpha 0.1

Índice

1. Introducción	2
2. Diagrama esquemático	2
3. Código del programa	2
4. Bitacoras Personales	8
4.1. Krapp Ramiro	8
4.1.1. 24/03/2022	8
4.1.2. 25/03/2022	8
4.1.3. 26/03/2022	8

El índice tiene hipervínculos incorporados! Toca en cada sección y automáticamente tu lector de pdfs te llevará a esa página

Tengo un [Repositorio en GitHub](https://github.com/KrappRamiro/traintorio)
<https://github.com/KrappRamiro/traintorio>

Introducción

Diagrama esquemático

Código del programa

— Código principal —

```
1  /*
2  TODO Hacer el registro de viajes de cada pasajero
3
4  */
5
6  #include <Arduino.h>
7  #include <iostream>
8  #include <vector>
9
10 /* Se usa std::vector en reemplazo de usar `using namespace std` por una muy
11 buena razón, y es que se evita el namespace pollution. Si no sabes qué es eso,
12 te recomiendo personalmente este post, es corto, sencillo, y bien explicado
13 para principiantes:
14 https://www.thecrazyprogrammer.com/2021/01/better-alternatives-for-using-namespace-std-in-c.html
15 */
16 using std::vector;
17
18 class Tren {
19     /* Si alguien se pregunta por qué las variables están en private,
20     la respuesta es muy sencilla:
21     Es porque no se desea que se modifiquen las variables de forma manual.
22     Esto es porque esa práctica es propensa a errores, ya que se podría introducir
23     un valor inadecuado y generar algún problema.
24
25     Por eso se usan funciones public, normalmente llamadas setters, que permiten
26     asignar y leer los valores, y que establecen un margen de valores seguros. */
27 private:
28     int speed = 0; // velocidad, en km/h
29     String serialNumber; // número de serie, que va a identificar al tren
30     String currentStation;
31     String trainType; // esta var se refiere si es a nafta, si es eléctrico, etc
32
33 public:
34     Tren(String serialNumber, String trainType)
35     {
36         this->serialNumber = serialNumber;
37         this->trainType = trainType;
38     }
39     // Para los getters tenía dos opciones, o retornaba un struct, o hacía una función
40     // para cada variable
41     int getSpeed()
42     {
43         return speed;
44     }
45     String getSerialNumber()
46     {
47         return serialNumber;
48     }
49     String getCurrentStation()
50     {
51         return currentStation;
52     }
53     String getTrainType()
54     {
55         return trainType;
```

```

56     }
57
58     void travelToStation(String stationName)
59     {
60         currentStation = stationName;
61         // TODO Hacer algo parecido con la funcion que tenes en Pasajero
62     }
63 };
64
65 class Persona {
66     // Esta clase sirve como padre para las clases Maquinista y Pasajero
67     // IDEA: Hacer que las personas puedan morir, y que se invalide la SUBE.
68     // Por ejemplo,         if (!persona.isAlive) {allowTransaction(false)}
69 private:
70     String name;
71     bool isAlive = true;
72     String dni;
73
74 public:
75     void kill()
76     {
77         isAlive = false;
78     }
79 };
80
81 class Maquinista : public Persona { // clase que hereda de Persona
82 private:
83     String name;
84     float salary;
85     int seniority; // el seniority se piensa con los años de antigüedad
86 public:
87 };
88
89 class Pasajero : public Persona { // clase que hereda de Persona
90 private:
91     String nombre;
92     int sube_id;
93     float sube_saldo;
94
95 public:
96     void travelToStation(String stationName)
97     {
98         // TODO hay que hacer la transaccion
99
100        /*
101        Como deberia ser esto? tendria que ser así:
102        1- Calcular distancia a la estacion
103        2- Cobrar 5 pesos por cada estacion
104
105        Para calcular la estación, lo que haría sería armar un vector de
106        estaciones, algo así:
107
108        ["temperley", "lomas de zamora", "banfield", "remedios de escalada", "etc"]
109
110        1 - Llamar a una funcion getCurrentStation() que retorne un String
111        de la estacion actual
112        2 - sabiendo la estacion actual, se podría hacer un getIndex()
113        https://www.geeksforgeeks.org/how-to-find-index-of-a-given-element-in-a-vector-in-cpp/
114        Entonces se haria un getIndex(estacionActual) - getIndex(estacionDestino),
115        y el resultado de esa operacion es la distancia entre las estaciones.
116
117        OJO: Esa operacion puede dar resultados negativos, por eso habria que guardarlo
118        en una variable, chequear si es negativa, y en ese caso pasarla a positivo
119
120        3 - Llamar a la funcion calcularPasaje(int price_per_estacion, int distance)
121
122        */
123

```

```

124     }
125 };
126
127 #include <MFRC522.h> //library responsible for communicating with the module RFID-RC522
128 #include <SPI.h> //library responsible for communicating of SPI bus
129 #define SS_PIN 21
130 #define RST_PIN 22
131 #define SIZE_BUFFER 18
132 #define MAX_SIZE_BLOCK 16
133 #define greenPin 12
134 #define redPin 32
135 // used in authentication
136 MFRC522::MIFARE_Key key;
137 // authentication return status code
138 MFRC522::StatusCode status;
139 // Defined pins to module RC522
140 MFRC522 mfrc522(SS_PIN, RST_PIN);
141
142 // reads data from card/tag
143 void readingData()
144 {
145     // prints the technical details of the card/tag
146     mfrc522.PICC_DumpDetailsToSerial(&(mfrc522.uid));
147
148     // prepare the key - all keys are set to FFFFFFFFh
149     for (byte i = 0; i < 6; i++)
150         key.keyByte[i] = 0xFF;
151
152     // buffer for read data
153     byte buffer[SIZE_BUFFER] = { 0 };
154
155     // the block to operate
156     byte block = 1;
157     byte size = SIZE_BUFFER; // authenticates the block to operate
158     status = mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A, block, &key, &(mfrc522.uid)); // line
159     ↪ 834 of MFRC522.cpp file
160     if (status != MFRC522::STATUS_OK) {
161         Serial.print(F("Authentication failed: "));
162         Serial.println(mfrc522.GetStatusCodeName(status));
163         digitalWrite(redPin, HIGH);
164         delay(1000);
165         digitalWrite(redPin, LOW);
166         return;
167     }
168
169     // read data from block
170     status = mfrc522.MIFARE_Read(block, buffer, &size);
171     if (status != MFRC522::STATUS_OK) {
172         Serial.print(F("Reading failed: "));
173         Serial.println(mfrc522.GetStatusCodeName(status));
174         digitalWrite(redPin, HIGH);
175         delay(1000);
176         digitalWrite(redPin, LOW);
177         return;
178     } else {
179         digitalWrite(greenPin, HIGH);
180         delay(1000);
181         digitalWrite(greenPin, LOW);
182     }
183
184     Serial.print(F("\nData from block ["));
185     Serial.print(block);
186     Serial.print(F("]: "));
187
188     // prints read data
189     for (uint8_t i = 0; i < MAX_SIZE_BLOCK; i++) {
190         Serial.write(buffer[i]);
191     }

```

```

191     Serial.println(" ");
192 }
193
194 void writingData()
195 {
196
197     // prints thecnical details from of the card/tag
198     mfrc522.PICC_DumpDetailsToSerial(&(mfrc522.uid));
199
200     // waits 30 seconds dor data entry via Serial
201     Serial.setTimeout(30000L);
202     Serial.println(F("Enter the data to be written with the '#' character at the end \n[maximum of 16
↵ characters]:"));
203
204     // prepare the key - all keys are set to FFFFFFFFh
205     for (byte i = 0; i < 6; i++)
206         key.keyByte[i] = 0xFF;
207
208     // buffer para almacenamiento dos dados que iremos gravar
209     // buffer for storing data to write
210     byte buffer[MAX_SIZE_BLOCK] = "";
211     byte block; // the block to operate
212     byte dataSize; // size of data (bytes)
213
214     // recover on buffer the data from Serial
215     // all characters before chacactere '#'
216     dataSize = Serial.readBytesUntil('#', (char*)buffer, MAX_SIZE_BLOCK);
217     // void positions that are left in the buffer will be filled with whitespace
218     for (byte i = dataSize; i < MAX_SIZE_BLOCK; i++) {
219         buffer[i] = ' ';
220     }
221
222     block = 1; // the block to operate
223     String str = (char*)buffer; // transforms the buffer data in String
224     Serial.println(str);
225
226     // authenticates the block to operate
227     // Authenticate is a command to hability a secure communication
228     status = mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A,
229         block, &key, &(mfrc522.uid));
230
231     if (status != MFRC522::STATUS_OK) {
232         Serial.print(F("PCD_Authenticate() failed: "));
233         Serial.println(mfrc522.GetStatusCodeName(status));
234         digitalWrite(redPin, HIGH);
235         delay(1000);
236         digitalWrite(redPin, LOW);
237         return;
238     }
239     // else Serial.println(F("PCD_Authenticate() success: "));
240
241     // Writes in the block
242     status = mfrc522.MIFARE_Write(block, buffer, MAX_SIZE_BLOCK);
243     if (status != MFRC522::STATUS_OK) {
244         Serial.print(F("MIFARE_Write() failed: "));
245         Serial.println(mfrc522.GetStatusCodeName(status));
246         digitalWrite(redPin, HIGH);
247         delay(1000);
248         digitalWrite(redPin, LOW);
249         return;
250     } else {
251         Serial.println(F("MIFARE_Write() success: "));
252         digitalWrite(greenPin, HIGH);
253         delay(1000);
254         digitalWrite(greenPin, LOW);
255     }
256 }
257

```

```

258 // menu to operation choice
259 int menu()
260 {
261     Serial.println(F("\nChoose an option:"));
262     Serial.println(F("0 - Reading data"));
263     Serial.println(F("1 - Writing data\n"));
264
265     // waits while the user does not start data
266     while (!Serial.available()) { };
267
268     // retrieves the chosen option
269     int op = (int)Serial.read();
270
271     // remove all characters after option (as \n per example)
272     while (Serial.available()) {
273         if (Serial.read() == '\n')
274             break;
275         Serial.read();
276     }
277     return (op - 48); // subtract 48 from read value, 48 is the zero from ascii table
278 }
279
280 void setup()
281 {
282     Serial.begin(9600);
283     SPI.begin(); // Init SPI bus
284     pinMode(greenPin, OUTPUT);
285     pinMode(redPin, OUTPUT);
286
287     digitalWrite(greenPin, HIGH);
288     digitalWrite(redPin, HIGH);
289     delay(500);
290     digitalWrite(greenPin, LOW);
291     digitalWrite(redPin, LOW);
292
293     // Init MFRC522
294     mfrc522.PCD_Init();
295     Serial.println("Approach your reader card...");
296     Serial.println();
297 }
298
299 void loop()
300 {
301     // Aguarda a aproximacao do cartao
302     // waiting the card approach
303     if (!mfrc522.PICC_IsNewCardPresent()) {
304         return;
305     }
306     // Select a card
307     if (!mfrc522.PICC_ReadCardSerial()) {
308         return;
309     }
310
311     // Dump debug info about the card; PICC_HaltA() is automatically called
312     // mfrc522.PICC_DumpToSerial(8(mfrc522.uid)); //call menu function and retrieve the desired option
313     int op = menu();
314
315     if (op == 0)
316         readingData();
317     else if (op == 1)
318         writingData();
319     else {
320         Serial.println(F("Incorrect Option!"));
321         return;
322     }
323
324     // instructs the PICC when in the ACTIVE state to go to a "STOP" state
325     mfrc522.PICC_HaltA();

```

```
326      // "stop" the encryption of the PCD, it must be called after communication with authentication, otherwise
      ↪ new communications can not be initiated
327      mfr522.PCD_StopCrypto1();
328  }
```

Bitacoras Personales

Krapp Ramiro

24/03/2022

- Comence creando un repositorio en github para subir todos los cambios del proyecto
- Cree un codigo en C++, para definir un sistema de clases. La idea es hacer una clase Tren, para que sirva de blueprint para todos los trenes, y una clase Persona, para que sea padre de otras dos clases, Maquinista y Pasajero. Al pasajero le voy a asignar una sube, y al maquinista le voy a asignar un salario y un seniority

25/03/2022

- Pienso implementar la sube con un sistema usando RFID
<https://randomnerdtutorials.com/security-access-using-mfrc522-rfid-reader-with-arduino/>
- La idea seria armar un sistema en el que cada usuario pueda tener un llavero RFID, y que asigne ese llavero RFID con una cuenta. Tambien necesito comprar los lectores para RFID. En total, tengo pensado comprar 2 lectores y 4 llaveros RFID. Por qué 2 lectores? Estaba pensando en asignar cada uno a una estación distinta. Por qué 4 llaveros? Estaba pensando en asignar cada uno a un pasajero distinto.
- Encontre que para en L^AT_EX dejar de tener problema con las url yendose fuera pantalla, puedo usar el paquete url con la opcion [hyphens], lo unico es que hay que cargar este paquete antes de hyperref. Esto es porque por defecto el paquete hyperref ya carga al paquete url <https://tex.stackexchange.com/questions/544671/option-clash-for-package-url-urlstyle>

Hacer las
urls mas chi-
cas con o
tiny

26/03/2022

- Encontre mucha documentacion del ESP32 y de proyectos con el RFID, la principal es esta:
- <https://arduinogetstarted.com/tutorials/arduino-rfid-nfc>
- https://olddocs.zerynth.com/latest/official/board.zerynth.doit_esp32/docs/index.html
- https://testzdoc.zerynth.com/reference/boards/doit_esp32/docs/
- <https://randomnerdtutorials.com/esp32-pinout-reference-gpios/>
- <https://randomnerdtutorials.com/getting-started-with-esp32/>
- Voy a usar el grafico de randomnerdtutorials, del link de getting-started..., el que incluye que pines son GPIO, me va a servir un montón. Para cuando quiera programar, solamente tengo que recordar que lo mejor es usar los GPIO del 13 al 33, y que mi DOIT ESP32 DevKit V1 es la version de 30 pines
- Decidi seguir el tutorial de este link <https://www.instructables.com/ESP32-With-RFID-Access-Control/>
- Hice andar el codigo durante un tiempo, grabe que funcionaba incluso, pero de repente dejo de funcionar, solamente me da un error: PCD_Authenticate () failed: Timeout in communication.