

The keypad: a lock with a secret code

Twelve keys on seven wires: the Keypad library reads what you press and Arduino checks the code. With light and sound.

By **Ada Intelisano** · informaticait.it technical team



Online guide,
always up to date

IN SHORT

- Pins from the left: rows to pins 2–5, columns to pins 8–10.
- Keypad library: `getKey` returns the key pressed.
- The digits are collected in a `String`; `#` confirms, `*` clears.
- No external resistors: the library uses the internal pull-ups.

Time: 30 minutes

Level: day twelve

Before: guides no. 35 and 38

Twelve keys like the ones on a phone, and a project everyone likes: a lock with a secret code. Type the digits, press `#`, and if the code is right the light turns on. If it is wrong, a low sound.

1 Get the parts ready

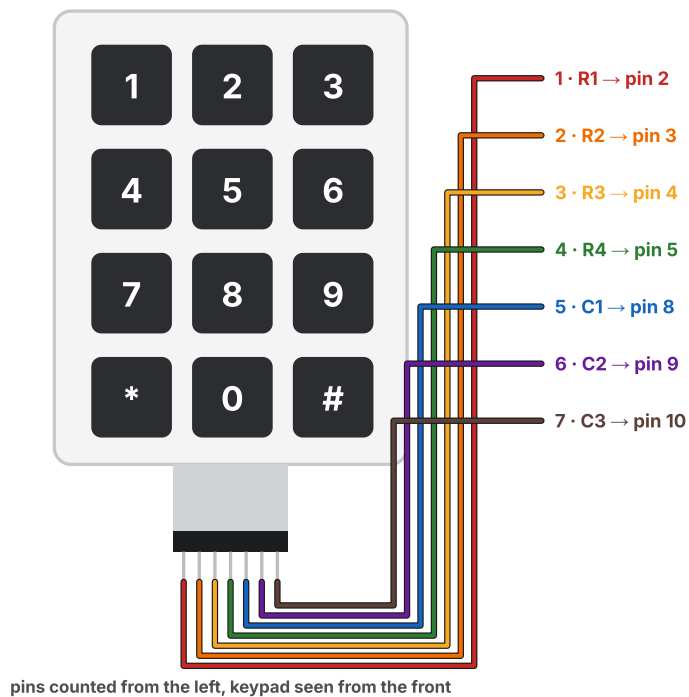
- a **3×4 membrane keypad** (thin and adhesive, with seven pins at the end of its tail)
- a **passive buzzer** (guide no. 38), Arduino UNO, seven male-to-male jumper wires

New word: matrix

The keys do not have one wire each: they are crossed over 4 rows and 3 columns. When you press a key, its row touches its column. Arduino tries the combinations one after another and works out which key is pressed: 7 wires for 12 keys.

2 Connect the seven pins

3×4 keypad: where each pin goes



informaticait.it

With the keypad in front of you, count the pins from the left: the first four are the rows and go to pins **2, 3, 4, 5**; the last three are the columns and go to pins **8, 9, 10**. The buzzer goes between pin **12** and GND. No resistors: the library takes care of it.

✓ Done?

Seven jumper wires in a row, in the right order: 2, 3, 4, 5, then 8, 9, 10.

3 Read the keys

From the Library Manager install **Keypad** by Mark Stanley and Alexander Brevig.

```
// The 3x4 keypad: prints the key you press in the Serial Monitor.
#include <Arduino.h>
#include <Keypad.h>

const byte ROWS = 4, COLS = 3;
char keys[ROWS][COLS] = {           // how the keypad is laid out
  {'1', '2', '3'},
  {'4', '5', '6'},
  {'7', '8', '9'},
  {'*', '0', '#'}};
};
byte rowPins[ROWS] = {2, 3, 4, 5};    // keypad pins 1-4 (with the ESP32: 13, 14, 27, 32)
byte colPins[COLS] = {8, 9, 10};      // keypad pins 5-7 (with the ESP32: 26, 25, 33)

Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

void setup() {
  Serial.begin(115200);
}

void loop() {
  char key = keypad.getKey(); // NO_KEY if you have not pressed anything
  if (key != NO_KEY) {
    Serial.println(key);
  }
}
```

New word: the key map

The keys table copies the layout of the keypad: 4 rows of 3. That way the library knows that row 1 with column 2 is the "2" key.

✓ What you should see

In the Serial Monitor (115200) every key you press appears.

If it doesn't work

- The wrong keys come out: two pins are swapped. Check the order from the left again.
- A whole row or a whole column does not respond: that jumper wire is not making contact.
- You have a different keypad (4×4, or from another maker): the pin order changes, check its datasheet.

4 The secret code

```
// The secret code: type the digits and press # to try it, * to clear. If it is right the
// small L light turns on.
#include <Arduino.h>
#include <Keypad.h>

const byte ROWS = 4, COLS = 3;
char keys[ROWS][COLS] = {
  {'1', '2', '3'},
  {'4', '5', '6'},
  {'7', '8', '9'},
  {'*', '0', '#'}
};

byte rowPins[ROWS] = {2, 3, 4, 5};           // with the ESP32: 13, 14, 27, 32
byte colPins[COLS] = {8, 9, 10};             // with the ESP32: 26, 25, 33
Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

const String SECRET = "1234";               // change it to your own code
const int LED = 13;                         // the small L light (with the ESP32: an LED on 23)
const int BUZZER = 12;                     // with the ESP32 write 4
String typed = "";                         // the digits pressed so far

void setup() {
  pinMode(LED, OUTPUT);
  Serial.begin(115200);
  Serial.println("Type the code and press #");
}

void loop() {
  char key = keypad.getKey();
  if (key == NO_KEY) return;                // nothing new: start the loop again

  tone(BUZZER, 2000, 30);                   // a small click for each key
  if (key == '*') {                         // clear
    typed = "";
    Serial.println("Cleared");
  } else if (key == '#') {                  // try the code
    if (typed == SECRET) {
      Serial.println("Right code!");
      digitalWrite(LED, HIGH);
      delay(3000);                          // open for 3 seconds
      digitalWrite(LED, LOW);
    } else {
      Serial.println("Wrong code");
      tone(BUZZER, 300, 600);                // a low, long sound
    }
    typed = "";
  } else {
    typed += key;                           // add the digit at the end
    Serial.print("*");                      // you only see an asterisk, like at the ATM
  }
}
```

New word: String

A variable that holds text. `typed += key` adds the digit just pressed at the end; `typed == SECRET` compares the two texts.

✓ What you should see

Type 1234 and press #: "Right code!" and the small L light turns on for 3 seconds. A wrong code makes a low sound; * clears.

5 Challenge yourself

Try it yourself

- A real lock in miniature: with the right code the servo from guide no. 41 turns 90 degrees.
- After three wrong codes the keypad locks for 30 seconds.
- Show the digits on the display from guide no. 53 as you type them.

6 And with the ESP32?

It works the same way: rows on GPIO **13, 14, 27, 32**, columns on **26, 25, 33**, buzzer on **4**. We avoid GPIO 12, which the ESP32 reads at power-up to decide how to start.

✓ What you learned

- a matrix keypad uses 7 wires for 12 keys;
- the Keypad library does the scanning: `getKey` gives you the key pressed;
- a `String` collects the digits and is compared with the code.

Sources: Adafruit, keypad 419 and "Matrix Keypad" guide · Keypad library by Mark Stanley and Alexander Brevig · [informaticait](http://informaticait.it) guides no. 38, 41 and 53