TO DEVELOPERS AND CURIOUS USERS

Renan Rocha Ribeiro

11, may of 2020

Foreword

The current source code of the HIGROTERM system is implemented using *functions* for everything. This may sound odd when one considers other widespread and well-known programming paradigms that may be more adequate to the work being done here, such as Object-oriented programming. In fact, it was only when this project was already mature that occurred to me that I should have implemented it using Object-oriented programming for better understand of interested users, developers, and collaborators. The use of such old-fashion functions to structure the system may be explained by two reasons:

- I am not fluent in Object-oriented programming;
- The HIGROTERM system was developed almost without intention, with small upgrades ranging along a wide period of time. So, it was not planned in the beginning to be released to the public as, maybe, a useful tool for others.

With that said, I am aware that refactoring the code to Object-oriented programming would contribute to others understanding the logic and structure I employed, and, ultimately, to a widespread of the system. I do not make a compromise to undertake this task, but once this code is completely open to the public, I do hope someone gets interested in doing this.

For others only looking to use the system, or willing to make only small upgrades and customizations, I hope the "function-ish" structure does not look so messy.

Source code structure

The basic structure of the source code is a core that runs continuously and performs two main tasks: (i) checks in which state the system is, i.e. which screen the user is accessing; (ii) handles the sampling process. For those familiar with Arduino, this core is, in fact, the ***void loop*** part of the source code.

The system currently contains 8 screens, and it mainly works from the so-called **Monitoring Table Screen.** This screen is the basic screen of the system, in which the user sees sensor's measurements, start and stop samplings, and access other screens. It is the screen in which the ***void loop*** runs upon.

The ***void loop*** continuously checks if buttons contained in the **Monitoring Table Screen** have been pressed by the user. The **Monitoring Table Screen** has three buttons that allows to execute the following tasks:

- Start a recording session;
- Stop a recording session;
- Open the **Configuration Screen**, that has its own buttons that may set system's variables or allow the user to access other screens.

Once ***void loop*** detects one of these buttons were pressed, the appropriate function is called by the system to perform these tasks. If a recording session has been started, the ***void loop*** starts to also handle this task by continuously checking if a new sample needs to be taken.

Once the **Configuration Screen** is accessed, the function used to draw it on the screen contains a loop function executing a task very similar to ***void loop***: continuously checks for buttons pressed, to either alter variable's values or to enter new screens. Once a new screen is accessed, the function used to draw it also contains a similar loop that perform these tasks also. In this way, the system navigates through various screens, once at a time, by calling its drawing functions.

Source file structure

Since almost all tasks of the system are executed by functions, it was decided to use a functionality supplied by the Arduino IDE that is dividing the source files (of the *.ino* extension) into various parts. When compiling, the IDE merges these files sequentially, based on their name's alphanumerical ordering.

In this way, intending to ease the understanding and navigation of other users throughout the code, the source file was divided in 37 small files, grouped with an alphanumerical codification based on what the functions contained in those files perform. Mostly, each file contains a single function, that lends its name to the file.

Codification	DESCRIPTION - What can you find in this series of files
A000	Main code of the system, in which all the other functions are called
A0X	All buttons definitions, used in all screens, are declared in this file
A1X	All functions related to the Initial Screen
A2X	All functions related to the Loading Screen
A3X	All functions related to the Monitoring Table Screen
A4X	All functions related to the Configuration Screen
A5X	All functions related to the SD card
A6X	All functions related to the DHT sensors
A7X	All functions related to the Set Clock Screen
A8X	All icons bitmaps are declared here
A9X	Miscellaneous functions, with general purposes, are declared here

Blank

(maybe not anymore...)