

A PharoThings Tutorial

Alex Oliveira

January 8, 2019

Copyright 2017 by Alex Oliveira.

The contents of this book are protected under the Creative Commons Attribution-ShareAlike 3.0 Unported license.

You are **free**:

- to **Share**: to copy, distribute and transmit the work,
- to **Remix**: to adapt the work,

Under the following conditions:

Attribution. You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).

Share Alike. If you alter, transform, or build upon this work, you may distribute the resulting work only under the same, similar or a compatible license.

For any reuse or distribution, you must make clear to others the license terms of this work. The best way to do this is with a link to this web page:

<http://creativecommons.org/licenses/by-sa/3.0/>

Any of the above conditions can be waived if you get permission from the copyright holder. Nothing in this license impairs or restricts the author's moral rights.



Your fair dealing and other rights are in no way affected by the above. This is a human-readable summary of the Legal Code (the full license):

<http://creativecommons.org/licenses/by-sa/3.0/legalcode>

Contents

Illustrations	ii
1 Lesson 11 - Building a Mini-Weather Station	1
1.1 What do we need?	1
1.2 Experimental theory	1
1.3 Experimental procedure	2
1.4 Creating the Thingspeak account	2
1.5 Creating the application	2
1.6 Creating the Superclass	4
1.7 Creating the subclass DisplayLCD	4
1.8 Creating the subclass PostData	5
1.9 Starting the application	5
1.10 Visualizing your data	5

Illustrations

1-1	ThingSpeak Channel Configuration.	3
1-2	Mini Weather Station code.	3
1-3	ThingSpeak Channel.	6



Lesson 11 - Building a Mini-Weather Station

In the previous lessons, we learned how to control LEDs, sensors, LCD displays and how to use OOP to create applications to control them. Now we will use everything that we learned to build a Mini-Weather Station.

1.1 What do we need?

In this lesson, we will use a setup with an LCD Display 1602 and BME280 sensor.

Components

- 1 Raspberry Pi connected to your network (wired or wireless)
- 1 Breadboard
- 1 BME280 temperature, humidity and pressure sensor
- 1 LCD Display 1602
- 1 Potentiometer (10K ohms)
- Jumper wires

1.2 Experimental theory

Before constructing any circuit, you must know the parameters of the components in the circuit, such as their operating voltage, operating circuit, etc.

In this lesson, we will get the temperature, pressure, and humidity using the BME280 sensor, show this information on the LCD Display each 1 second and send this data to a Cloud Server every 60 seconds.

1.3 Experimental procedure

Connect the sensor and LCD Display on your breadboard as we did in the previous lessons.

1.4 Creating the Thingspeak account

Before we start to create an application, let's create our account on the website Thingspeak. We will go to use this website to receive and store the data.

Follow this tutorial to create your account and your channel:

<https://roboindia.com/tutorials/thingspeak-setup>

Once you've created your channel, you need to enable three channels on it. Let's use Field1 to Temperature, Field2 to Humidity and Field3 to Pressure. You can see in Picture 1-1 how your channel will look like.

You can test your channel by simulating sending some data to it. Let's for example send the values:

- 25 to Temperature field (field1);
- 56 to Humidity field (field2);
- and 1012 to Pressure field (field3).

Just copy the following URI and passed in your web browser, changing YOUR-WRITE-API-KEY to your ThingSpeak write API code. You can find it in the API Keys tab.

https://api.thingspeak.com/update?api_key=YOUR-WRITE-API-KEY&field1=25&field2=56&field3=1012

So check you channel to see if it was updated with this values:

<https://thingspeak.com/channels/YOUR-CHANNEL-ID>

1.5 Creating the application

The first step lets create a Superclass to initialize and install all devices that we will use. So let's create 2 subclasses to do the actions. The first will display this information on the LCD and the second will send the data to the cloud. Your final code will seem like the Picture 1-2.

Channels ▾
Apps ▾
Community
Support ▾

PharoThings Monitor - INRIA Alex's Office

Channel ID: 562010
Author: [allexoliveira](#)
Access: Public

Getting temperature, humidity, and pressure from a BME280 sensor using PharoThings running in Raspberry Pi 3 B+.

Private View
Public View
Channel Settings
Sharing
API Keys
Data Import / Export

Channel Settings

Percentage complete 50%

Channel ID 562010

Name

Description

Field 1	<input type="text" value="Temperature"/>	<input checked="" type="checkbox"/>
Field 2	<input type="text" value="Humidity"/>	<input checked="" type="checkbox"/>
Field 3	<input type="text" value="Pressure"/>	<input checked="" type="checkbox"/>

Figure 1-1 ThingSpeak Channel Configuration.

×

▢

WeatherStation

Scoped

Variables

weath

Last Modified Methods

Configurations

Work

PharoThings-MiniWeather

WeatherStation

DisplayLCD

PostData

History Navigator

-- all --

access

initialization

humidity

initialize

pressure

temperature

Hier.

Class

Com.

```

Object subclass: #WeatherStation
  instanceVariableNames: 'sensor lcd'
  classVariableNames: ''
  package: 'PharoThings-MiniWeatherStation'

```

Figure 1-2 Mini Weather Station code.

1.6 Creating the Superclass

```
Object subclass: #WeatherStation
  instanceVariableNames: 'sensor lcd'
  classVariableNames: ''
  package: 'PharoThings-MiniWeatherStation'
```

Creating the initialize method

```
initialize
  lcd := (RpiBoard3B current) installDevice: PotLCD1602Device new.
  sensor := (RpiBoard3B current) installDevice: PotBME280Device new.
```

Creating access methods

```
humidity
  ^((sensor readParameters at: 3) printShowingDecimalPlaces: 2)
  asString.

pressure
  ^((sensor readParameters at: 2) printShowingDecimalPlaces: 2)
  asString.

temperature
  ^((sensor readParameters at: 1) printShowingDecimalPlaces: 2)
  asString.
```

1.7 Creating the subclass DisplayLCD

```
WeatherStation subclass: #DisplayLCD
  instanceVariableNames: 'lcdprocess'
  classVariableNames: ''
  package: 'PharoThings-MiniWeatherStation'

  lcdStart
    |text|
    lcdprocess := [ [
      text := 'Temp: ',self temperature,'\H:',self humidity,' P:',self
      pressure.
      lcd home.
      lcd message: text.
      (Delay forSeconds: 1) wait.
    ] repeat ] forkNamed: 'lcdprocess'.

  lcdStop
    lcdprocess terminate.
    lcd clear.
```


1.8 Creating the subclass PostData

```

WeatherStation subclass: #PostData
  instanceVariableNames: 'apiKey postProcess'
  classVariableNames: ''
  package: 'PharoThings-MiniWeatherStation'

  apiKey
    ^apiKey

  apiKey: anString
    apiKey := anString .

  dataStart
    |url uri |
    url := 'https://api.thingspeak.com/update'.
    postProcess := [ [
      uri := url,'?api_key=',self apiKey,'&field1=',self
      temperature,'&field2=',self humidity,'&field3=',self pressure.
      ZnClient new get: uri.
      (Delay forSeconds: 60) wait.
    ] repeat ] forkNamed: 'postprocess'.

  dataStop
    postProcess terminate.

```

1.9 Starting the application

To start the application, we need to start the two subclasses. To start the LCD application run this in the remote playground:

```
(DisplayLCD new) lcdStart.
```

and to start to send the data to the Cloud, use this, replacing to your apiKey of Thingspeak:

```
(PostData new) apiKey:'F1MKEG7PJ44930L8'; dataStart.
```

1.10 Visualizing your data

To see if your channel is receiving the data of your PharoThings, just access your channel and check if the data is being updated:

<https://thingspeak.com/channels/YOUR-CHANNEL-ID>

You will see some grafics like the Picture 1-3.

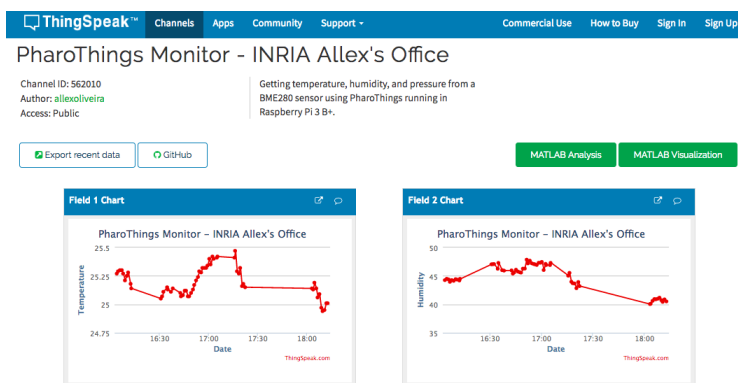


Figure 1-3 ThingSpeak Channel.