

SIT210 Project Artefact

Smart computer temperature sensor

OverView

- Background:

When computer working, the GPU, CPU and the other parts inside the computer will generate heat and it is very obvious on high-performance PC. If computer work at extremely high temperature it may cause damage to itself, and it is necessary to find a way to prevent the computer overheat. To improve the efficiency of heat radiation I build a water cooling system to my PC.

- Problem Statment:

After building up the system, I don't know can it work better or not because sometimes I sit around the PC but I will feel warm and hot. I think there are two possible reasons why I feel warm and hot, the first reason is hot air remains inside the case, and the other reason is the hot air is sent out from the radiator. TO find which hypothesis is correct I will use an embedded system to help me.

- Requirement:

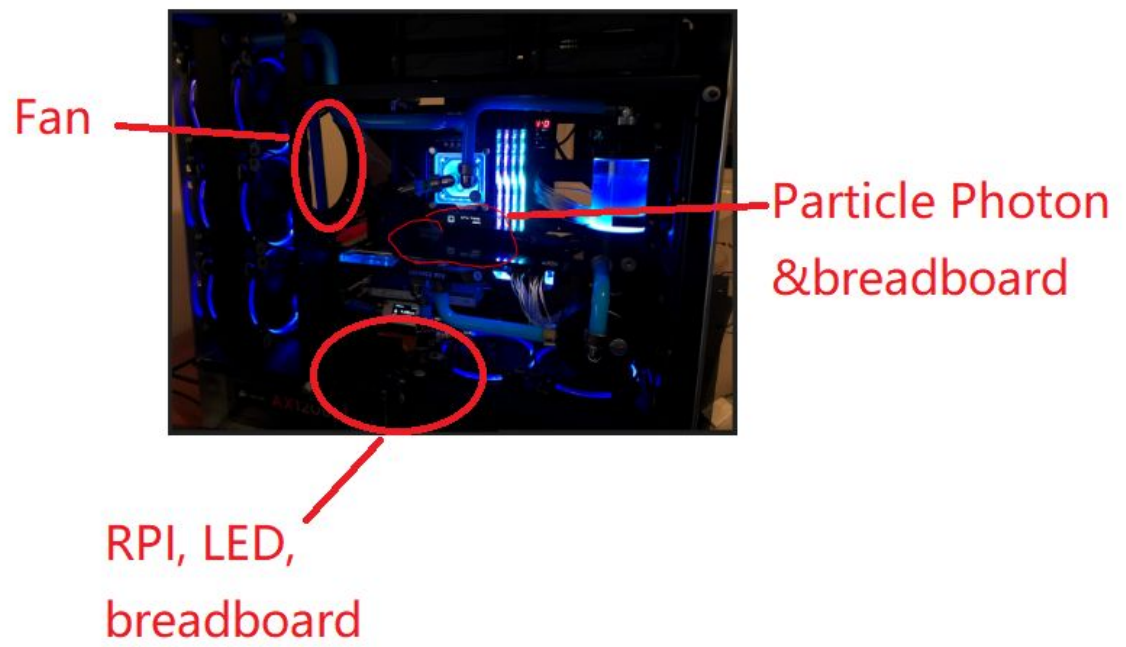
- Hardware requirement:

- Particle Photon
- Raspberry Pi
- Breadboard *2
- LED *3(red, orange, green)
- Resistors between 220 Ohms and 1000 Ohms *2
- USB to micro USB cable *2 (one for Photon one for RPI)
- Power source for USB cable (in here I will you the PC USB port)
- Temperature sensor(TMP102 using I2C communication)

- 120mm*120mm PC fan
 - Jumper wire
 - Software requirement:
 - Particle Agent(install in RPI)
 - Particle Web IDE
 - IFTTT
 - ThingSpeak
- Design Principles:

Particle Photon will be used to detect the temperature and publish the corresponding event. RPI will be used to subscribe to the event from Photon return the corresponding LED colour and fan speed. I will use a webhook connect to Thingspeak to output how the temperature change into a diagram, on the other hand, I will use an IFTTT to give the notification to the user if the temperature is in high level.
- Prototype Architecture:

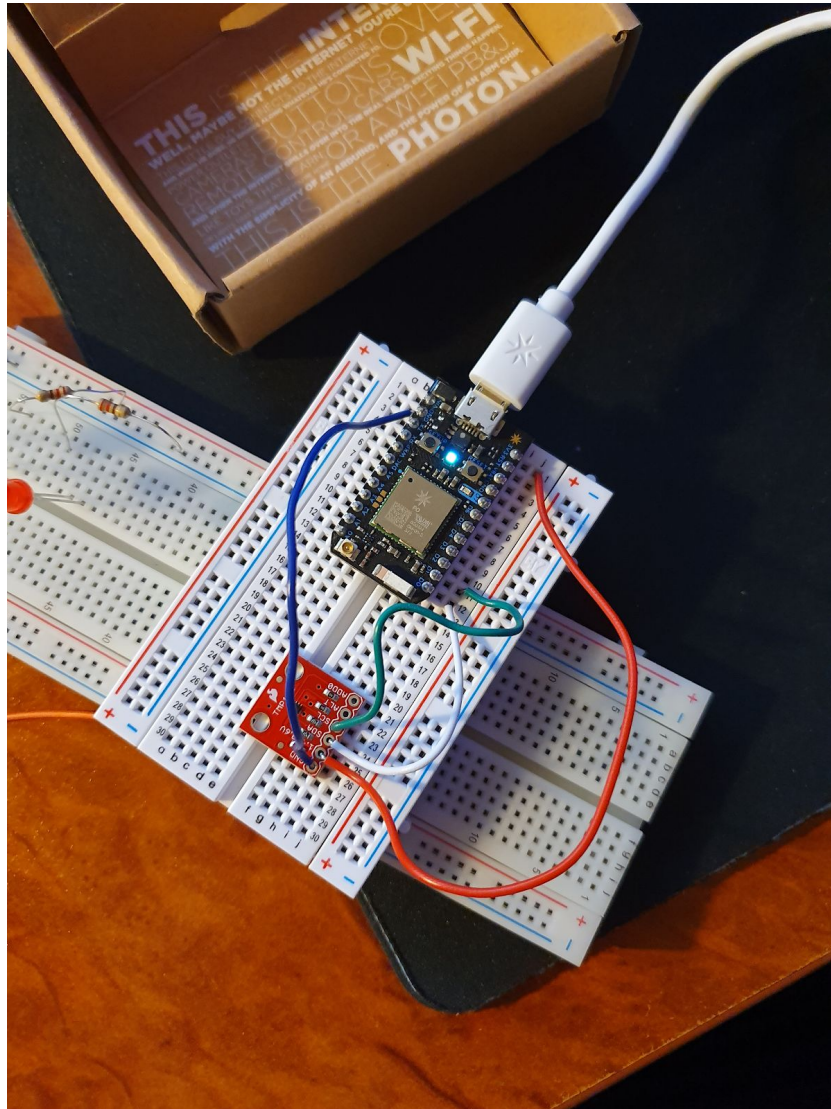
All the hardware will be placed in the PC case.

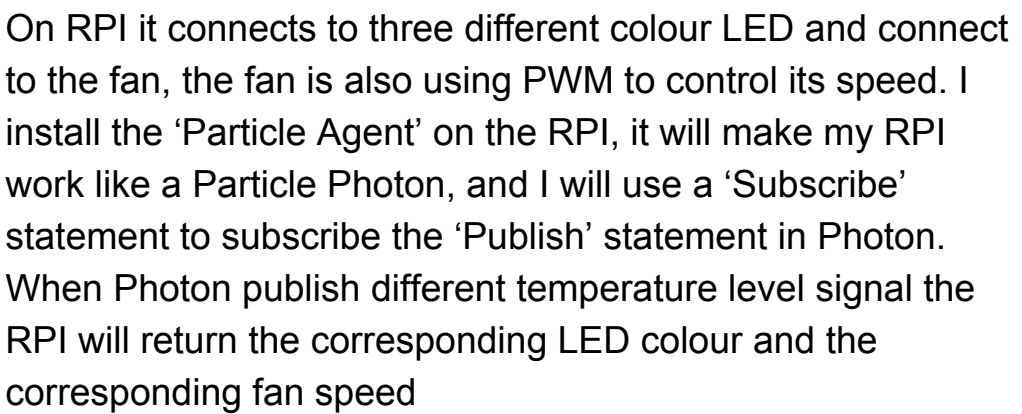


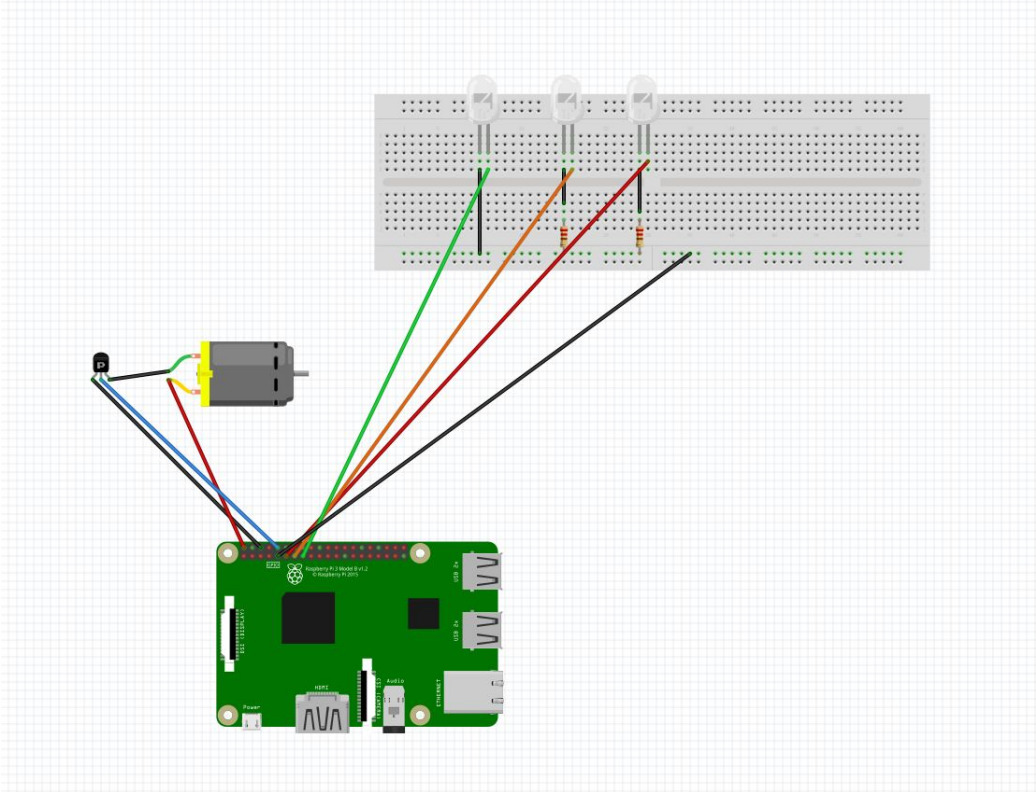


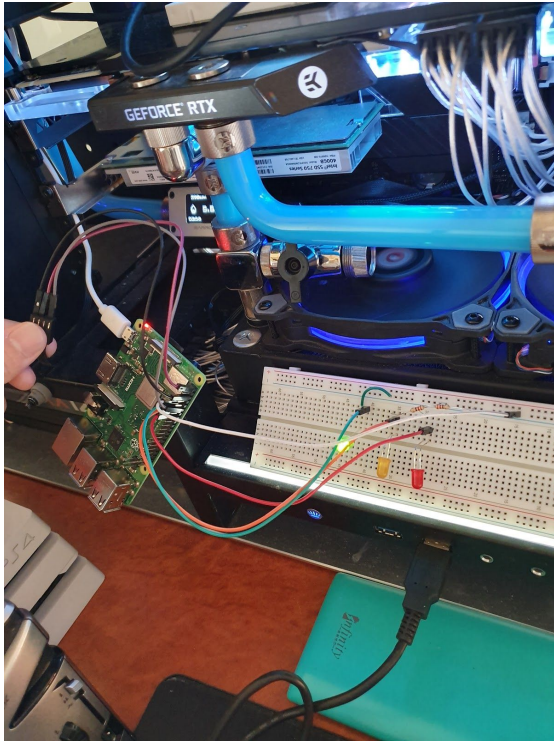
On Particle Photon I will just connect the temperature sensor to detect the temperature inside the PC case, and I will use Particle Web IDE design the code, in the Particle Web IDE I will use the 'Publish' statement to publish the temperature

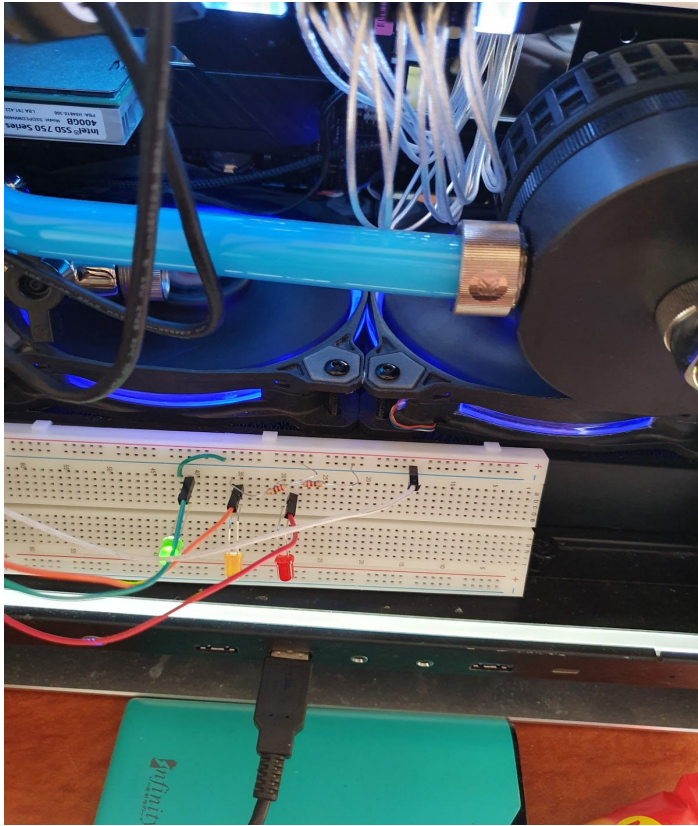
value and the level of the temperature. On the other hand, I will use a webhook which connect to the Thingspeak and output the temperature to a diagram let user able to see how's the temperature change inside the PC case. I also use an IFTTT connects to the phone app, when Photon publishes the Event call ' Temperature' and the content of the event is ' HIGH' it will send to message to user phone to remind them their PC is in high temperature.











On the software setup:

- compile the code of RPI and the Photon on Particle Web IDE and send them.
- setup webhook using Thingspeak
 - firstly set up the Particle Console page first

[Integrations](#) > Edit Integration

WEBHOOK BUILDER CUSTOM TEMPLATE

[Read the Particle webhook guide](#)

Event Name ⓘ

temp

URL ⓘ

https://api.thingspeak.com/update

Request Type ⓘ

POST

Request Format ⓘ

Web Form

Device ⓘ

zzh_eric

▼ **Advanced Settings**

For information on dynamic data that can be sent in any of the fields below, please visit [our docs](#).

FORM FIELDS ⓘ

api key can found in Thingspeak

☐ Default ☒ Custom

api_key

>

[REDACTED]

×

field1

>

{{PARTICLE_EVENT_VALUE}}

×

+ ADD ROW

QUERY PARAMETERS ⓘ

○

○ then switch to Thingspeak

Access: Private

Private View

Public View

Channel Settings

Sharing

API Keys

Data Import / Export

Write API Key

Key

[Redacted Key]

Generate New Write API Key

Read API Keys

Key

[Redacted Key]

Note

[Empty Note Field]

Save Note

Delete API Key

Generate New Read API Key

Help

API keys enable you to w
keys are auto-generated

API Keys Setting

- **Write API Key:** Use
been compromise
- **Read API Keys:** Us
feeds and charts.
read key for the cl
- **Note:** Use this fiel
add notes to keep

API Requests

Update a Channel F

GET https://api.t

Get a Channel Feed

GET https://api.t

Get a Channel Field

GET https://api.t

-
- setting up the Thingspeak channel

Channel Settings

Percentage complete 50%

Channel ID 733904

Name

Description

Field 1 ☒

Field 2 ☐

Field 3 ☐

Field 4 ☐

Field 5 ☐

Field 6 ☐

Field 7 ☐

Field 8 ☐

Metadata

Tags

(Tags are comma separated)

Help

Channels store all the data that a ThingSpeak application collects. Each channel has eight fields that can hold any type of data, plus three fields for location data and a status data. Once you collect data in a channel, you can use ThingSpeak apps to visualize it.

Channel Settings

- **Channel Name:** Enter a unique name for the ThingSpeak channel.
- **Description:** Enter a description of the ThingSpeak channel.
- **Field#:** Check the box to enable the field, and enter a field name. Each channel can have up to 8 fields.
- **Metadata:** Enter information about channel data, including JSON, XML, etc.
- **Tags:** Enter keywords that identify the channel. Separate tags with commas.
- **Link to External Site:** If you have a website that contains information about your ThingSpeak channel, specify the URL.
- **Show Channel Location:**
 - **Latitude:** Specify the latitude position in decimal degrees. For example, the latitude of the city of London is 51.5072.
 - **Longitude:** Specify the longitude position in decimal degrees. For example, the longitude of the city of London is -0.1275.
 - **Elevation:** Specify the elevation position in meters. For example, the elevation of the city of London is 35.052.
- **Video URL:** If you have a YouTube™ or Vimeo® video that displays your channel information, specify the full path of the video URL.
- **Link to GitHub:** If you store your ThingSpeak code on GitHub®, specify the repository URL.

Using the Channel

You can get data into a channel from a device, website, or another ThingSpeak channel. You can then visualize data and transform it using [ThingSpeak Apps](#).

See [Tutorial: ThingSpeak and MATLAB](#) for an example of measuring dew point.

- Using IFTTT set up an IFTTT trigger

Configure



If `zzh_eric` published `TEMP_LEVEL`, then Send a notification from the IFTTT app

77/140

[View activity log](#)



New event published

This Trigger fires when an interesting event comes from a particular device. Send events using `Particle.publish`.

If (Event Name)

`TEMP_LEVEL`

Fill in your published event name; ex: monitoring a washing machine? Event Name = `Wash_Status`

is (Event Contents)

`HIGH`

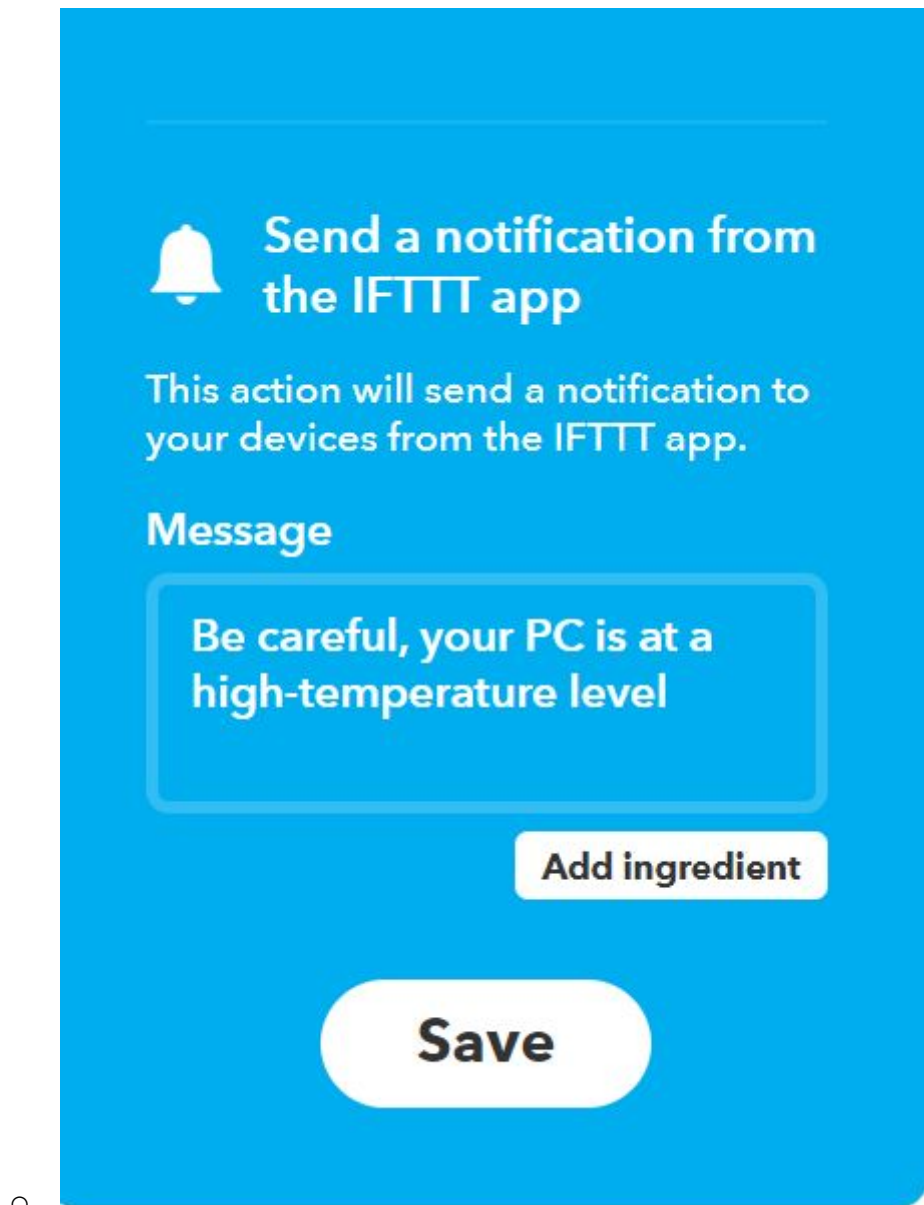
The contents of the published event, "Data"; ex: monitoring a washing machine? Event Contents = `Done`

Device Name or ID

`zzh_eric`



An optional id for a particular device



- Link to prototype code on Github:
<https://github.com/zzh900101/210-11.2-project.git>
<https://github.com/zzh900101/210-11.2-project>

Testing:

- I will test both hardware(led, fan) and software(Thingspeak, IFTTT) individually. To test all the things I will change the lowest temp level(15 degrees celcius) and the highest level(20 degrees celcius) in the code to show how the fan

speed, led colour change and the IFTTT work. The Thingspeak will always work when the Photon is running.

- I record a video about testing the project both hardware and the software:

<https://www.youtube.com/watch?v=vKLL1XDeIYc>

- When testing Photon will publish event 'HIGH' and the RPI will return led in red and the speed of the fan will turn up
- The software part will see the Thingspeak graph and the IFTTT activity log
 - IFTTT

在线... WeChat網頁版 rpg c# 少女前线-16LAB研... photon UniHub Jens | SIT2



Applet ran

May 16 - 3:51 PM



If zzh_eric published TEMP_LEVEL, then Send a notification from the IFTTT app

zzh_eric published TEMP_LEVEL

▼ Show details



Applet ran

May 16 - 3:51 PM



If zzh_eric published TEMP_LEVEL, then Send a notification from the IFTTT app

zzh_eric published TEMP_LEVEL

▼ Show details



Applet ran

May 16 - 3:51 PM



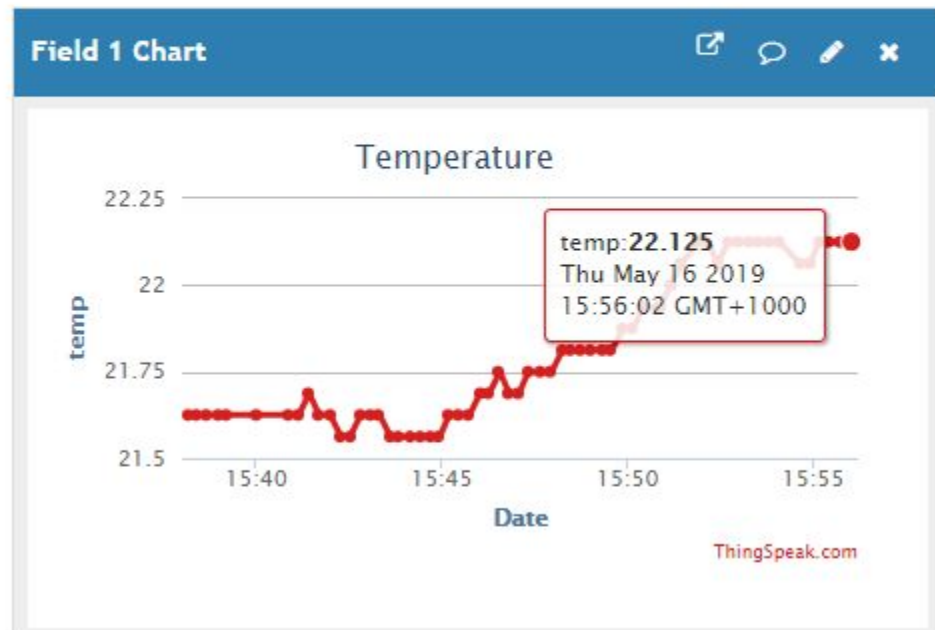
If zzh_eric published TEMP_LEVEL, then Send a notification from the IFTTT app

- Thingspeak

Created: about a month ago

Last entry: less than a minute ago

Entries: 12974



User Manual:

1. Connect the hardware(RPI, Photon, Breadboard, LEDs, fan, sensor) correctly.
2. In RPI install 'Particle agent'

<https://docs.particle.io/reference/discontinued/particle-agent/>

3. Setting up the Thingspeak channel and copy the API key

[Private View](#) [Public View](#) [Channel Settings](#) [Sharing](#) [API Keys](#) [Data Import / Export](#)

Channel Settings

Percentage complete 50%

Channel ID 733904

Name

Description

Field 1	<input type="text" value="temp"/>	<input checked="" type="checkbox"/>
Field 2	<input type="text"/>	<input type="checkbox"/>
Field 3	<input type="text"/>	<input type="checkbox"/>
Field 4	<input type="text"/>	<input type="checkbox"/>
Field 5	<input type="text"/>	<input type="checkbox"/>
Field 6	<input type="text"/>	<input type="checkbox"/>
Field 7	<input type="text"/>	<input type="checkbox"/>
Field 8	<input type="text"/>	<input type="checkbox"/>

Metadata

Tags

(Tags are comma separated)

Help

Channels store all the data that a ThingSpeak application collects. Each channel includes eight fields that can hold any type of data, plus three fields for location data and one for status data. Once you collect data in a channel, you can use ThingSpeak apps to analyze and visualize it.

Channel Settings

- **Channel Name:** Enter a unique name for the ThingSpeak channel.
- **Description:** Enter a description of the ThingSpeak channel.
- **Field#:** Check the box to enable the field, and enter a field name. Each ThingSpeak channel can have up to 8 fields.
- **Metadata:** Enter information about channel data, including JSON, XML, or CSV data.
- **Tags:** Enter keywords that identify the channel. Separate tags with commas.
- **Link to External Site:** If you have a website that contains information about your ThingSpeak channel, specify the URL.
- **Show Channel Location:**
 - **Latitude:** Specify the latitude position in decimal degrees. For example, the latitude of the city of London is 51.5072.
 - **Longitude:** Specify the longitude position in decimal degrees. For example, the longitude of the city of London is -0.1275.
 - **Elevation:** Specify the elevation position meters. For example, the elevation of the city of London is 35.052.
- **Video URL:** If you have a YouTube™ or Vimeo® video that displays your channel information, specify the full path of the video URL.
- **Link to GitHub:** If you store your ThingSpeak code on GitHub®, specify the GitHub repository URL.

Using the Channel

You can get data into a channel from a device, website, or another ThingsSpeak channel. You can then visualize data and transform it using [ThingSpeak Apps](#).

See [Tutorial: ThingSpeak and MATLAB](#) for an example of measuring dew point from a

Access: Private

Private View

Public View

Channel Settings

Sharing

API Keys

Data Import / Export

Write API Key

Key

[Redacted Key]

Generate New Write API Key

Read API Keys

Key

[Redacted Key]

Note

[Empty Note Field]

Save Note

Delete API Key

Generate New Read API Key

Help

API keys enable you to w
keys are auto-generated

API Keys Setting

- **Write API Key:** Use
been compromise
- **Read API Keys:** Us
feeds and charts.
read key for the cl
- **Note:** Use this fiel
add notes to keep

API Requests

Update a Channel F

GET <https://api.t>

Get a Channel Feed

GET <https://api.t>

Get a Channel Field

GET <https://api.t>

O

4. Setting up the webhook in Particle Console and put the API key in the advanced setting

[Integrations](#) > Edit Integration

WEBHOOK BUILDER CUSTOM TEMPLATE

[Read the Particle webhook guide](#)

Event Name ⓘ

temp

URL ⓘ

https://api.thingspeak.com/update

Request Type ⓘ

POST

Request Format ⓘ

Web Form

Device ⓘ

zzh_eric

▼ **Advanced Settings**

For information on dynamic data that can be sent in any of the fields below, please visit [our docs](#).

FORM FIELDS ⓘ

api key can found in Thingspeak

☐ Default ☒ Custom

api_key

>

[Redacted]

×

field1

>

{{PARTICLE_EVENT_VALUE}}

×

+ ADD ROW

QUERY PARAMETERS ⓘ

5. Setting up the IFTTT

Configure



If `zzh_eric` published `TEMP_LEVEL`, then Send a notification from the IFTTT app

77/140

[View activity log](#)



New event published

This Trigger fires when an interesting event comes from a particular device. Send events using `Particle.publish`.

If (Event Name)

`TEMP_LEVEL`

Fill in your published event name; ex: monitoring a washing machine? Event Name = `Wash_Status`

is (Event Contents)

`HIGH`

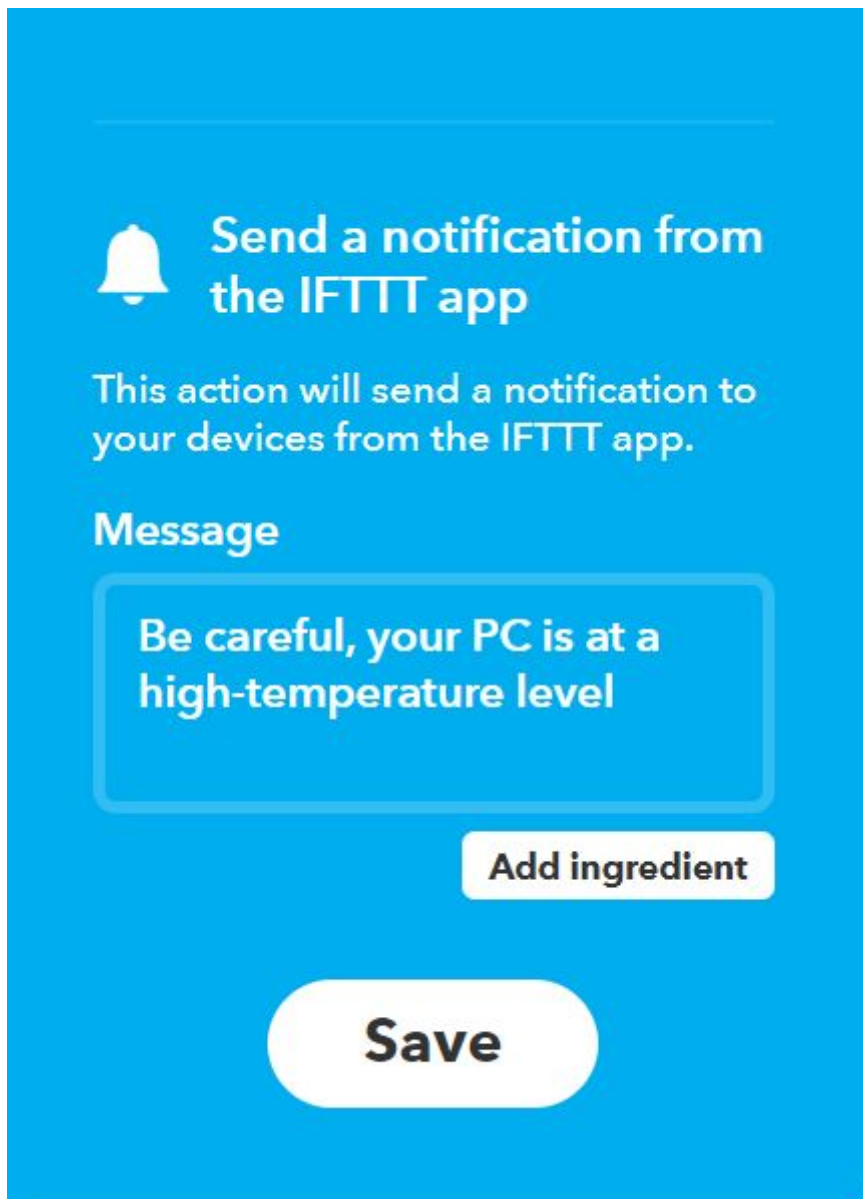
The contents of the published event, "Data"; ex: monitoring a washing machine? Event Contents = `Done`

Device Name or ID

`zzh_eric`



An optional id for a particular device



6. Using Particle Web IDE sends the code to the Particle Photon and the RPI.

7. Finished

- Conclusion:

Through designing and preparing the project I have learned from it, such as using the I2C communication on Particle Photon(I just only know how to use it in RPI python editor) and know how to use a Photon control the RPI to do something(using Particle Agent). When doing the project I also meet some problem such as the temperature sensor will return the incorrect value and I found this is the connection

problem, and I also need to know how to use PWM on Particle Photon(I have learned how to use on RPI python editor but not using Particle Web IDE, because the programming language they use is different, one is C++ and the other is Python).

If I have the second chance I will try to add the control command and HTML to the Photon to allow the user to control the fan speed) I also want to try to add a 12V power supply to the Fan because its normal work environment is not in 5V, add a power supply to the fan can let the fan works better and the hot air will be transferred more efficiently.