

<https://www.halvorsen.blog>



LabVIEW and ThingSpeak

Hans-Petter Halvorsen

Contents

- Introduction
- ThingSpeak
- LabVIEW
- LabVIEW Examples
 - Write Single Value to ThingSpeak
 - Read Single Value from ThingSpeak
 - Datalogging – Read Data from a Temperature Sensor and Log Data to ThingSpeak
 - Read Historical Data from ThingSpeak



Introduction

Introduction

- In this Tutorial we will see how we can use ThingSpeak in combination with LabVIEW
- ThingSpeak is an IoT service that lets you collect and store sensor data in the cloud
- LabVIEW is a popular graphical programming environment
- Lots of LabVIEW Examples will be provided for writing Data to ThingSpeak and Reading Data from ThingSpeak

<https://www.halvorsen.blog>

 ThingSpeak™



ThingSpeak

Hans-Petter Halvorsen

[Table of Contents](#)

ThingSpeak

- ThingSpeak is an IoT service that lets you collect and store sensor data in the cloud and develop Internet of Things applications.
- ThingSpeak is free for small non-commercial projects
- In addition, they offer different types of licenses where you pay a monthly fee
- ThingSpeak is owned by MathWorks, the same vendor that develop the MATLAB software
- <https://thingspeak.com>

ThingSpeak

Here you see an example of how Data can be presented in the ThingSpeak Web page

<https://thingspeak.com>

The screenshot displays the ThingSpeak web interface for a channel named "temperature". The top navigation bar includes "Channels", "Apps", "Devices", and "Support". The channel information shows "Channel ID: [redacted]", "Author: [redacted]", and "Access: Public". The channel is currently in "Public View".

Below the channel information, there are buttons for "Add Visualizations", "Add Widgets", "Export recent data", and "More Information". On the right side, there are buttons for "MATLAB Analysis" and "MATLAB Visualization".

The "Channel Stats" section indicates the channel was created "4 years ago", has a last entry "less than a minute ago", and contains 242 entries.

Four data visualization charts are displayed in a 2x2 grid:

- Field 1 Chart:** "Office Temperature" showing a fluctuating red line graph of temperature in Celsius over time (15:00 to 15:10).
- Field 2 Chart:** "Outdoor Temperature" showing a blank graph with the y-axis labeled "Temperature @ Building (C)" and the x-axis labeled "Date".
- Field 3 Chart:** "TMP36 Temperature" showing a fluctuating red line graph of temperature over time (11:45 to 12:30).
- Field 4 Chart:** "Work" showing a blank graph with the y-axis labeled "kp" and the x-axis labeled "Date".

ThingSpeak

- It works with Arduino, Raspberry Pi and MATLAB (premade libraries and APIs exists).
- But it should work with all kind of Programming Languages, since it uses a **REST API** and **HTTP**.
- **LabVIEW** has built-in **HTTP Client** functions that you can use for this purpose
- **MQTT** API also available

ThingSpeak – Channel Settings

Channel ID: [redacted]
Author: hansha
Access: Public

temperature

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

Channel Settings

Percentage complete 65%

Channel ID [redacted]

Name Work

Description

Field 1 Office Temperature [C]

Field 2 Temperature B Buildin

Field 3 Tout

Field 4 Kp

Field 5 Ti

Field 6 SP

Field 7 Field7

Field 8 Field8

Help

Channels store all the data that a ThingSpeak app sends to the server. Each channel has up to eight fields that can hold any type of data, plus the status data. Once you collect data in a channel, you can visualize it.

Channel Settings

- **Percentage complete:** Calculated based on the number of fields that are complete. Enter the name, description, location, URL, video, and tags to complete your channel.
- **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.

You can set up different Channels in ThingSpeak.

Each Channel can have up to 8 Fields.

You can have up to 4 different Channels for the Free License.

ThingSpeak - REST API

The screenshot shows the ThingSpeak API management interface for a channel named 'temperature'. The interface includes a navigation bar with tabs for 'Private View', 'Public View', 'Channel Settings', 'Sharing', 'API Keys', and 'Data'. The 'API Keys' tab is active, showing two sections: 'Write API Key' and 'Read API Keys'. The 'Write API Key' section has a 'Key' input field and a 'Generate New Write API Key' button. The 'Read API Keys' section has a 'Key' input field, a 'Note' text area, and 'Save Note' and 'Delete API Key' buttons. A 'Help' section on the right provides 'API Keys Settings' and 'API Requests'.

Channel ID: [redacted] | temperature

Author: [redacted]

Access: Public

Private View Public View Channel Settings Sharing API Keys Data

Write API Key

Key [redacted]

Generate New Write API Key

Read API Keys

Key [redacted]

Note [redacted]

Save Note Delete API Key

Add New Read API Key

Key needed to Write Data to the Channel

Key needed to Read Data from the Channel

Help

API Keys Settings

- **Write API Key:** Use this key to write data to a channel. If you feel your key has been compromised, click **Generate New Write API Key**.
- **Read API Keys:** Use this key to allow other people to view your private channel feeds and charts. Click **Generate New Read API Key** to generate an additional read key for the channel.
- **Note:** Use this field to enter information about channel read keys. For example, add notes to keep track of users with access to your channel.

API Requests

Write a Channel Feed

```
GET https://api.thingspeak.com/update?api_key=[redacted]&field=[redacted]
```

Read a Channel Feed

```
GET https://api.thingspeak.com/channels/[redacted]/feeds.json?results=2
```

Read a Channel Field

```
GET https://api.thingspeak.com/channels/[redacted]/fields/1.json?results=
```

REST API – Write Data

Use your standard Web Browser (e.g., Microsoft Edge, or Google Chrome) and enter the following:

```
https://api.thingspeak.com/update?api_key=XXXXXXXXXXXXXXXXXXXX&field1=20
```



Your **Write API Key**

Field Number 1-8

Value

REST API – Write Multiple Fields

Use your standard Web Browser (e.g., Microsoft Edge, or Google Chrome) and enter the following:

`https://api.thingspeak.com/update?api_key=XXXXXXXXXXXXXXXXXXXX&field1=21&field2=24`



Your **Write API Key**

Field + Value

Field + Value Etc.

REST API – Read Data

Use your standard Web Browser (e.g., Microsoft Edge, or Google Chrome) and enter the following:

Data Format (JSON or XML)

`https://api.thingspeak.com/channels/xxxxxx/fields/1.json?results=10`



```
{ "channel": { "id": "1", "name": "Work", "latitude": "0.0", "longitude": "0.0", "field1": "Office Temperature [C]", "field2": "Temperature B  
Building [C]", "field3": "Tou", "field4": "Kp", "field5": "T1", "field6": "Sp", "field7": "Field7", "field8": "field8", "created_at": "2017-05-  
30T11:41:00Z", "updated_at": "2021-09-09T10:59:27Z", "last_entry_id": 21, "feeds": [{"created_at": "2021-09-  
08T12:54:04Z", "entry_id": 12, "field1": null}, {"created_at": "2021-09-08T13:03:54Z", "entry_id": 13, "field1": null}, {"created_at": "2021-09-  
09T09:27:34Z", "entry_id": 14, "field1": "20.00"}, {"created_at": "2021-09-09T09:34:38Z", "entry_id": 15, "field1": null}, {"created_at": "2021-  
09-09T09:35:35Z", "entry_id": 16, "field1": "18.00"}, {"created_at": "2021-09-09T10:46:11Z", "entry_id": 17, "field1": "0.00"},  
{ "created_at": "2021-09-09T10:48:45Z", "entry_id": 18, "field1": "21"}, {"created_at": "2021-09-09T11:06:32Z", "entry_id": 19, "field1": "20"},  
{ "created_at": "2021-09-09T11:09:46Z", "entry_id": 20, "field1": "21"}, {"created_at": "2021-09-09T11:17:08Z", "entry_id": 21, "field1": "22"}]}
```

Your Channel ID

Field Number

Resulting JSON String with Data

Number of Data Points, e.g., 1 for only the last value, 10 for the last 10 values, etc.

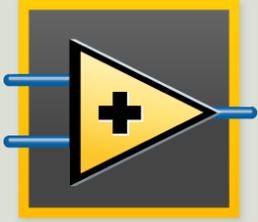
REST API – Read Data (JSON)

<https://api.thingspeak.com/channels/xxxxxx/fields/1.json?results=10>

```
{"channel":{"id":"xxxxxx","name":"Work","latitude":"0.0","longitude":"0.0","field1":"Office Temperature [C]","field2":"Temperature B Building [C]","field3":"Tout","field4":"Kp","field5":"Ti","field6":"SP","field7":"Field7","field8":"Field8","created_at":"2017-05-30T11:41:00Z","updated_at":"2021-09-09T10:59:27Z","last_entry_id":21},  
"feeds":  
  [{"created_at":"2021-09-08T12:54:04Z","entry_id":12,"field 1":null},  
  {"created_at":"2021-09-08T13:03:54Z","entry_id":13,"field 1":null},  
  {"created_at":"2021-09-09T09:27:34Z","entry_id":14,"field 1":"20.00"},  
  {"created_at":"2021-09-09T09:34:38Z","entry_id":15,"field 1":null},  
  {"created_at":"2021-09-09T09:35:35Z","entry_id":16,"field 1":"18.00"},  
  {"created_at":"2021-09-09T10:46:11Z","entry_id":17,"field 1":"0.00"},  
  {"created_at":"2021-09-09T10:48:45Z","entry_id":18,"field 1":"25"},  
  {"created_at":"2021-09-09T11:06:32Z","entry_id":19,"field 1":"20"},  
  {"created_at":"2021-09-09T11:09:46Z","entry_id":20,"field 1":"21"},  
  {"created_at":"2021-09-09T11:17:08Z","entry_id":21,"field 1":"22"}  
]}
```

Values

We need to parse the JSON string in order to get the actual Values



LabVIEW

Hans-Petter Halvorsen

[Table of Contents](#)

LabVIEW

- LabVIEW is a graphical programming language
- LabVIEW is created by National Instruments (now just NI)
- LabVIEW has powerful features for Simulation, Control and DAQ Applications

Want to learn LabVIEW?

- LabVIEW Resources:

<https://www.halvorsen.blog/documents/programming/labview/labview>

- LabVIEW in Automation:

https://www.halvorsen.blog/documents/teaching/courses/labview_automation.php

- LabVIEW Community Edition - free for non-commercial use:

<https://www.ni.com/en-no/shop/labview/select-edition/labview-community-edition.html>

LabVIEW and ThingSpeak



LabVIEW HTTP Client palette

The image shows three palettes from the LabVIEW software interface. The first palette, titled "Data Communication", contains various communication-related icons. The "Protocols" icon in this palette is circled in red. A red arrow points from this icon to the second palette, titled "Protocols", which lists various network protocols. In this palette, the "HTTP Client" icon is circled in red. A second red arrow points from the "HTTP Client" icon to the third palette, titled "HTTP Client", which contains specific HTTP methods and options. The "HTTP Client" palette includes icons for "Open Handle", "GET", "HEAD", "Close Handle", "PUT", "POST", "POST Multipart", "DELETE", "Security", and "Headers".

Data Communication

↑ Search Customize

Shared Variable Network Streams Local Variable Global Variable

Queue Operations Synchronization DataSocket Protocols

Actor Framework

Install WebSockets ... EPICS Modbus OPC UA

RTI DDS Toolkit

Protocols

↑ Search Customize

TCP UDP Serial IrDA Bluetooth

SMTP Email HTTP Client FTP WebDAV Wait for Configured ...

HTTP Client

↑ Search Customize

Open Handle GET HEAD Close Handle

PUT POST POST Multipart DELETE

Security Headers

We can use the built-in **HTTP Client** palette in order to create LabVIEW applications that communicate with the **ThingSpeak REST API**.



LabVIEW Examples

LabVIEW Examples

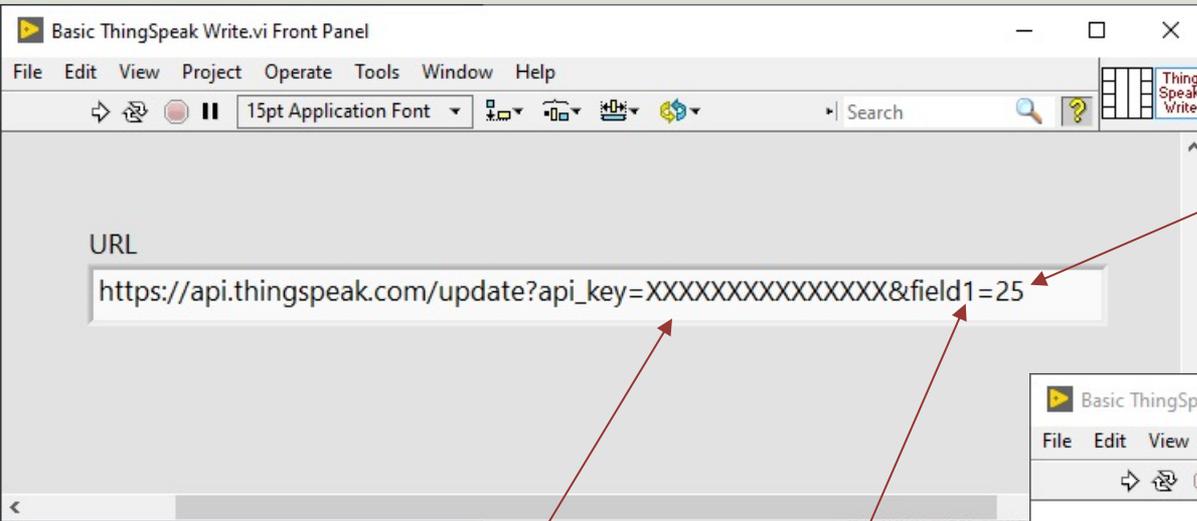
- Write Single Value to ThingSpeak
- Read Single Value from ThingSpeak
- Datalogging – Read Data from a Temperature Sensor and Log Data to ThingSpeak
- Read Historical Data from ThingSpeak



LabVIEW Examples

Write Single Value to ThingSpeak

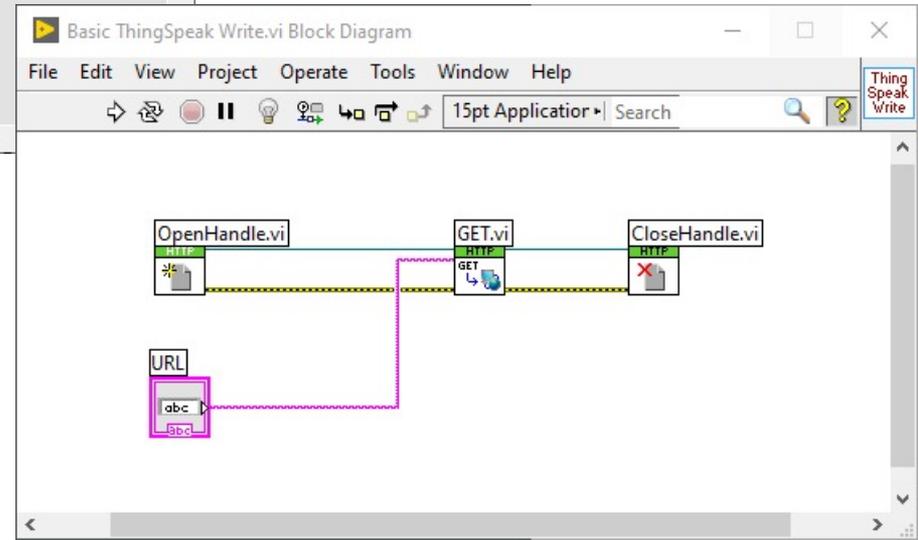
Write Single Field Value



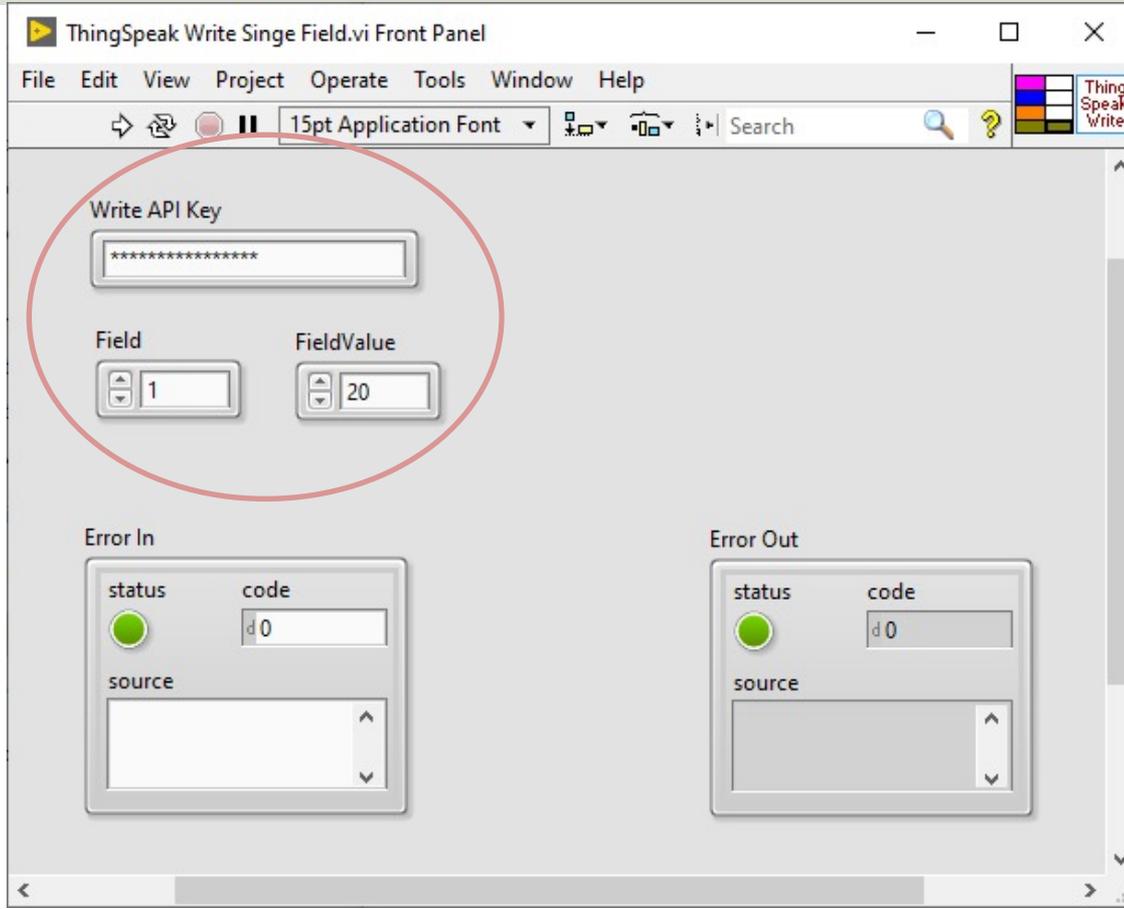
Value

Your Write API Key

Field Number 1-8

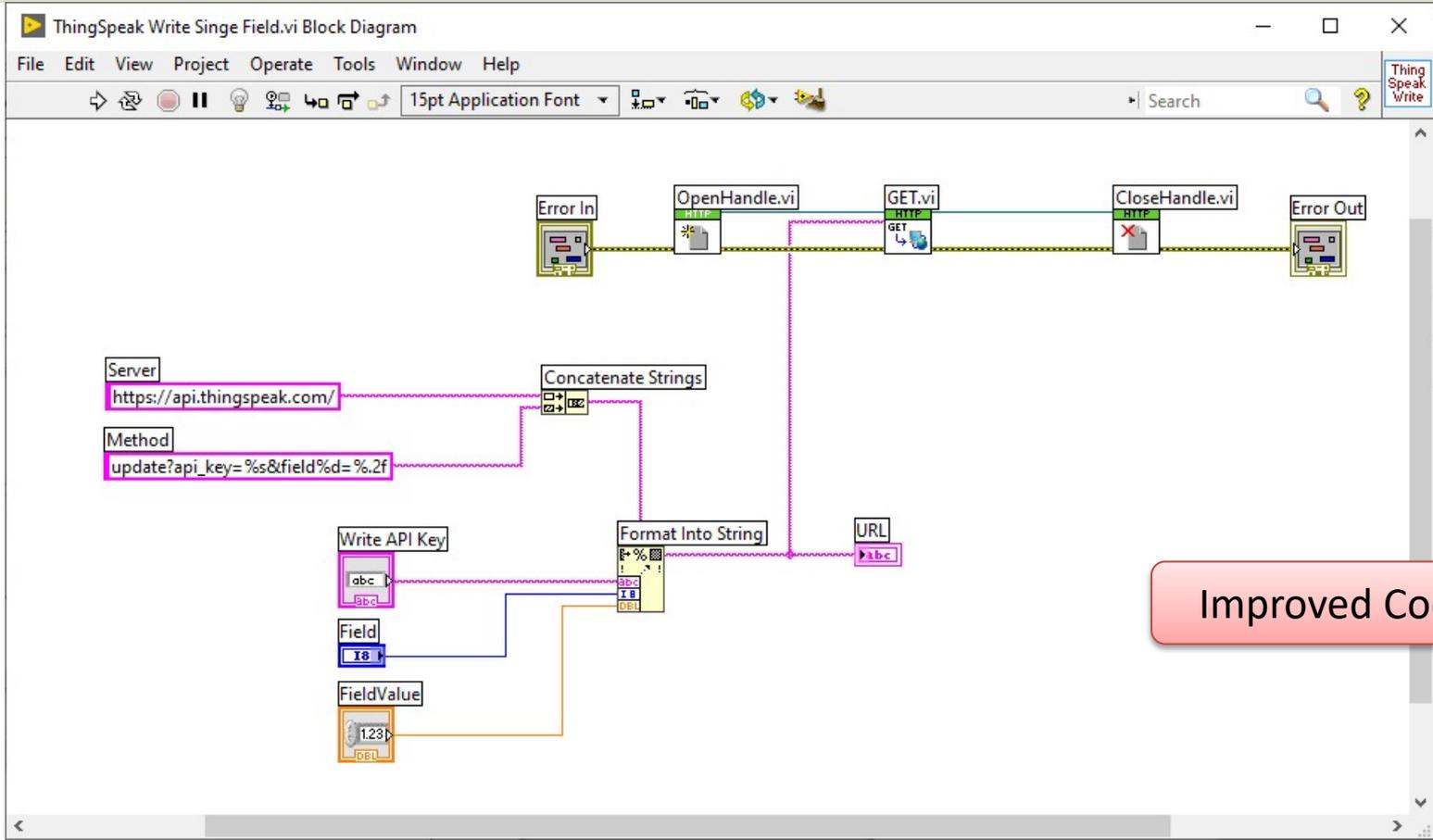


Write Single Field Value

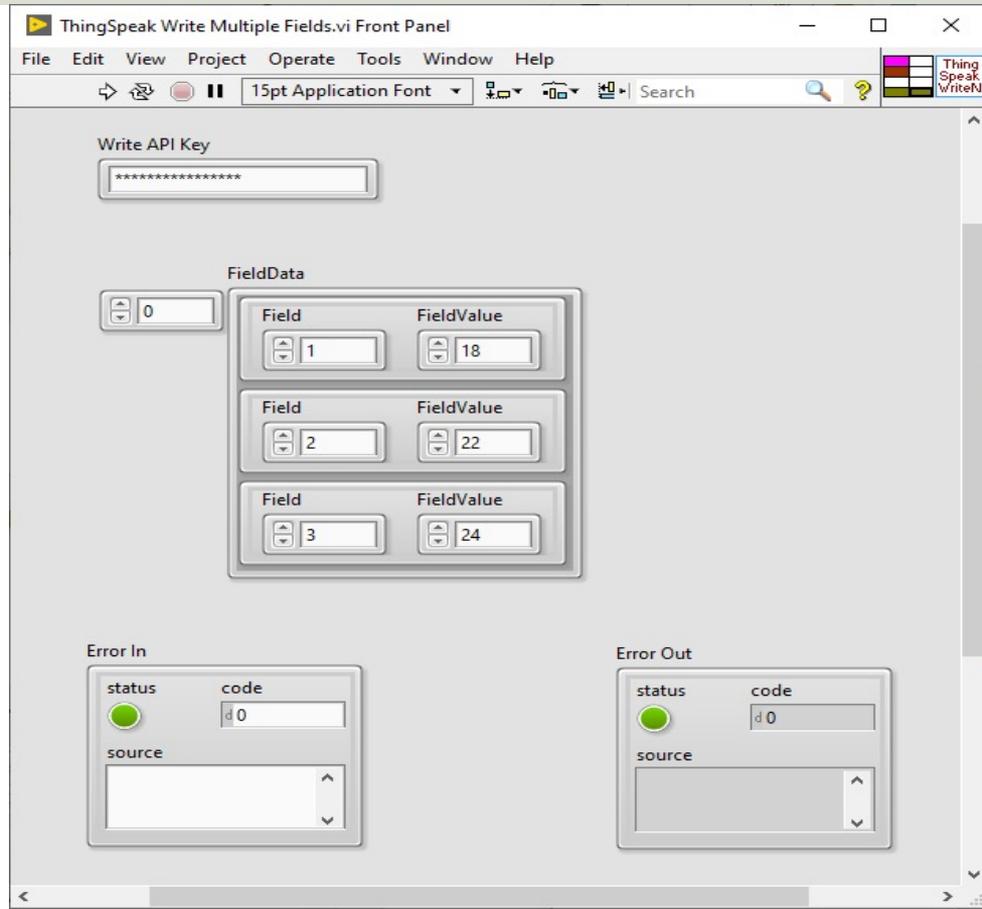


Improved Code

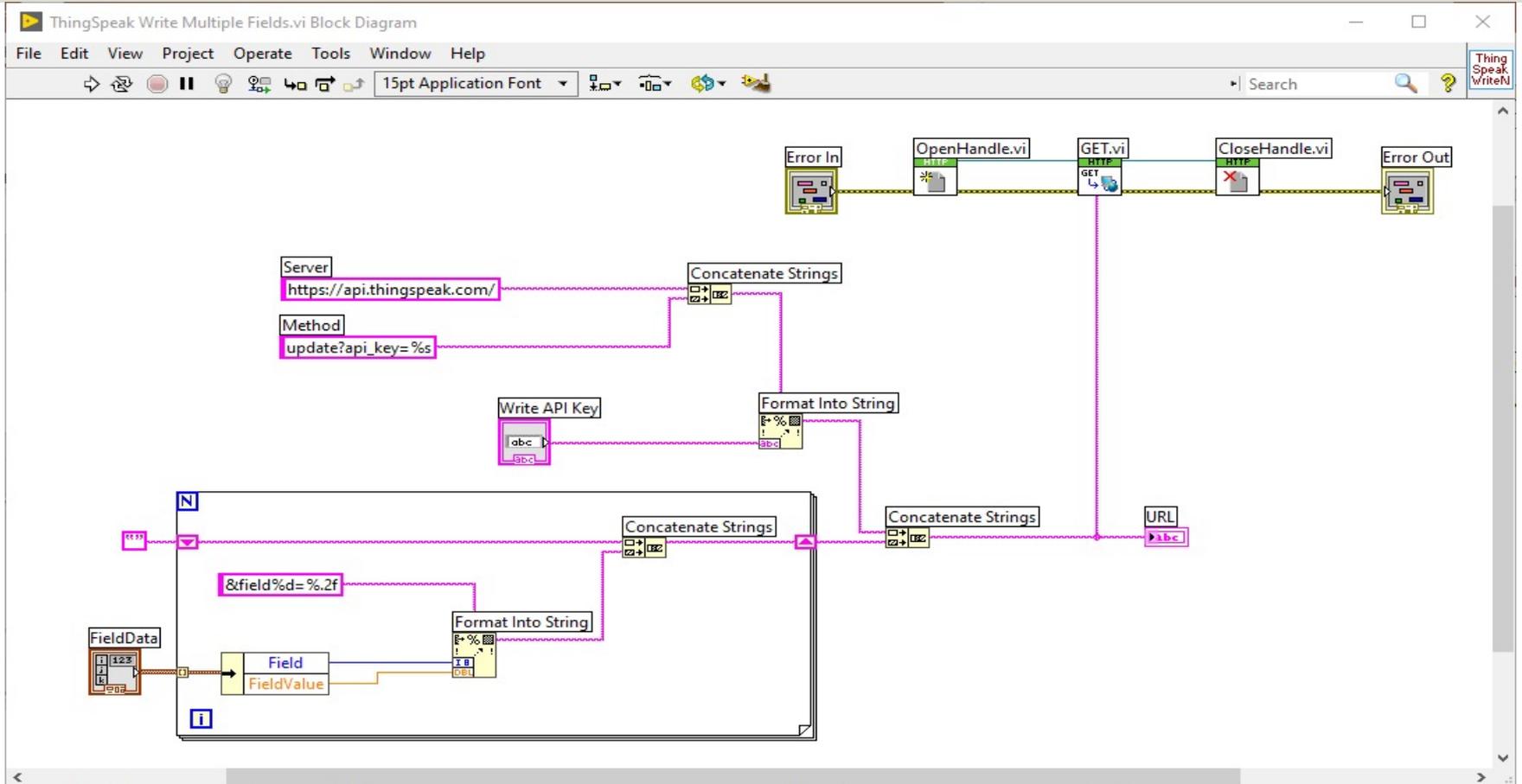
Block Diagram



Write Single Value to Multiple Fields



Block Diagram

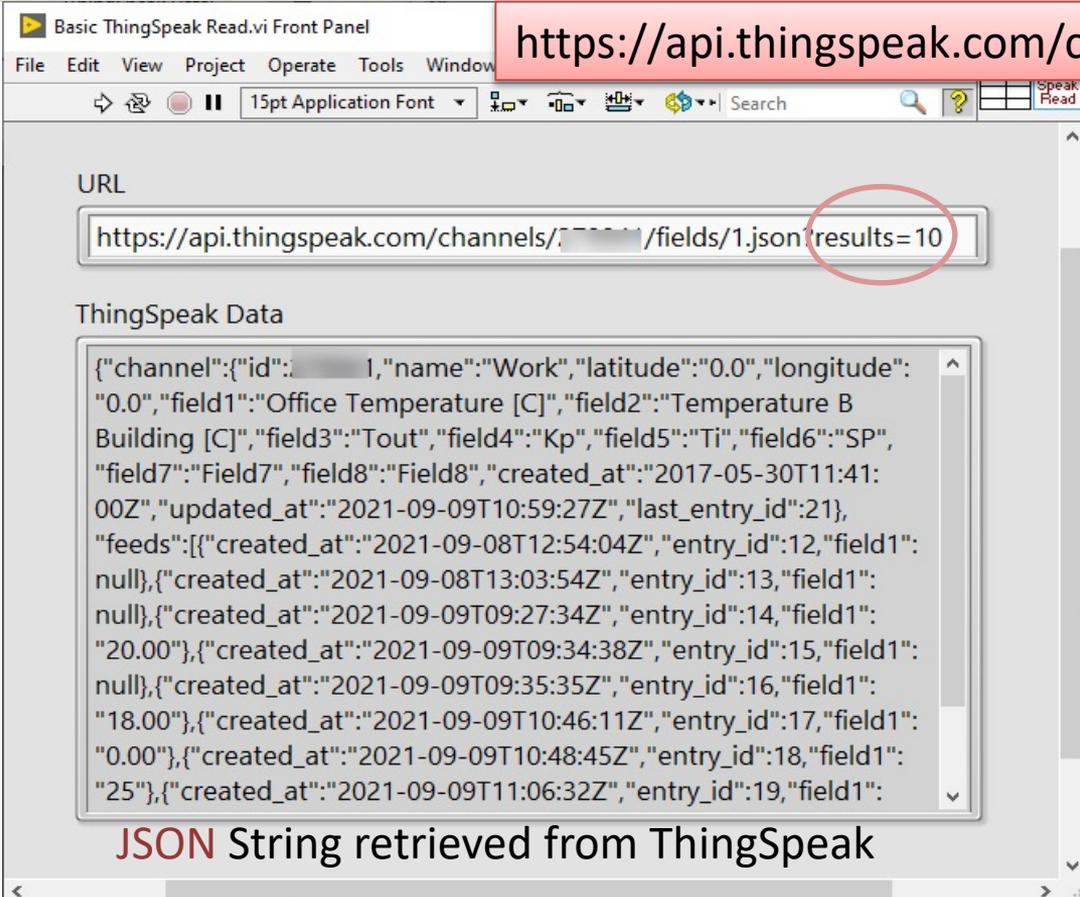




LabVIEW Examples

Read Single Value from ThingSpeak

Read Single Field Value(s)



Basic ThingSpeak Read.vi Front Panel

File Edit View Project Operate Tools Window

15pt Application Font

URL

<https://api.thingspeak.com/channels/xxxxxx/fields/1.json?results=10>

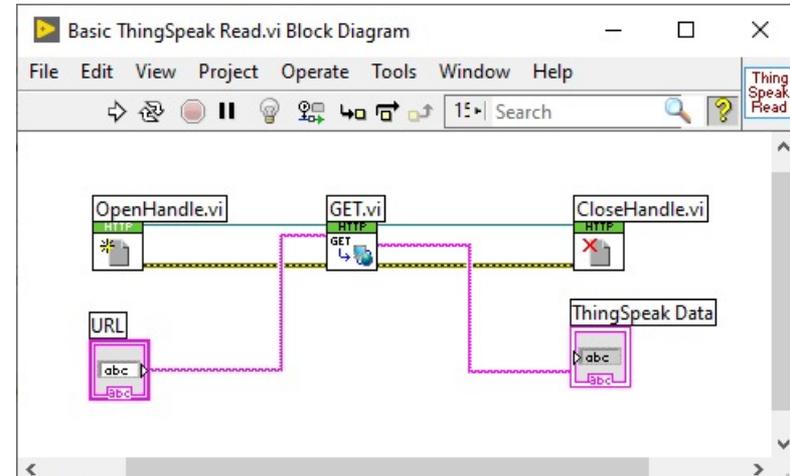
ThingSpeak Data

```
{ "channel": { "id": "xxxxxx", "name": "Work", "latitude": "0.0", "longitude": "0.0", "field1": "Office Temperature [C]", "field2": "Temperature B Building [C]", "field3": "Tout", "field4": "Kp", "field5": "Ti", "field6": "SP", "field7": "Field7", "field8": "Field8", "created_at": "2017-05-30T11:41:00Z", "updated_at": "2021-09-09T10:59:27Z", "last_entry_id": 21, "feeds": [ { "created_at": "2021-09-08T12:54:04Z", "entry_id": 12, "field1": null }, { "created_at": "2021-09-08T13:03:54Z", "entry_id": 13, "field1": null }, { "created_at": "2021-09-09T09:27:34Z", "entry_id": 14, "field1": "20.00" }, { "created_at": "2021-09-09T09:35:35Z", "entry_id": 15, "field1": null }, { "created_at": "2021-09-09T10:46:11Z", "entry_id": 16, "field1": "18.00" }, { "created_at": "2021-09-09T10:48:45Z", "entry_id": 17, "field1": "0.00" }, { "created_at": "2021-09-09T11:06:32Z", "entry_id": 18, "field1": "25" } ] }
```

JSON String retrieved from ThingSpeak

<https://api.thingspeak.com/channels/xxxxxx/fields/1.json?results=10>

ThingSpeak Data in JSON Format



Basic ThingSpeak Read.vi Block Diagram

File Edit View Project Operate Tools Window Help

15 Search

OpenHandle.vi GET.vi CloseHandle.vi

URL

ThingSpeak Data

```
graph LR; URL[URL] --> GET[GET.vi]; GET --> Data[ThingSpeak Data];
```

Read Single Field Value(s)

<https://api.thingspeak.com/channels/xxxxxx/fields/1.xml?results=10>

Basic ThingSpeak Read - XML.vi Front Panel

File Edit View Project Operate Tools Window

15pt Application Font

URL

`https://api.thingspeak.com/channels/ /fields/1.xml?results=10`

ThingSpeak Data

```
<?xml version="1.0" encoding="UTF-8"?>
<channel>
  <id type="integer">279941</id>
  <name>Work</name>
  <latitude type="decimal">0.0</latitude>
  <longitude type="decimal">0.0</longitude>
  <field1>Office Temperature [C]</field1>
  <field2>Temperature B Building [C]</field2>
  <field3>Tout</field3>
  <field4>Kp</field4>
  <field5>Ti</field5>
  <field6>SP</field6>
  <field7>Field7</field7>
```

XML String retrieved from ThingSpeak

ThingSpeak Data in XML Format

Basic ThingSpeak Read.vi Block Diagram

File Edit View Project Operate Tools Window Help

15 Search

OpenHandle.vi GET.vi CloseHandle.vi

URL abc

ThingSpeak Data abc

Read Single Field Value

ThingSpeak Read Single Field.vi Front Panel

File Edit View Project Operate Tools Window Help

15pt Application Font Search

ChannelID

Field
2

URL
https://api.thingspeak.com/channels/././fields/2.xml?results=1

body

```
<?xml version="1.0" encoding="UTF-8"?>
<channel>
  <id type="integer">././</id>
  <name>Work</name>
  <latitude type="decimal">0.0</latitude>
  <longitude type="decimal">0.0</longitude>
  <field1>Office Temperature [C]</field1>
  <field2>Temperature B Building [C]</field2>
  <field3>Tout</field3>
  <field4>Kp</field4>
  <field5>Ti</field5>
  <field6>SP</field6>
  <field7>Field7</field7>
  <field8>Field8</field8>
  <created-at type="dateTime">2017-05-30T11:41:00Z</created-at>
  <updated-at type="dateTime">2021-09-09T10:59:27Z</updated-at>
  <last-entry-id type="integer">21</last-entry-id>
  <feeds type="array">
    <feed>
      <created-at type="dateTime">2021-09-09T11:17:08Z</created-at>
```

Error In

status code
source

Error Out

status code
source

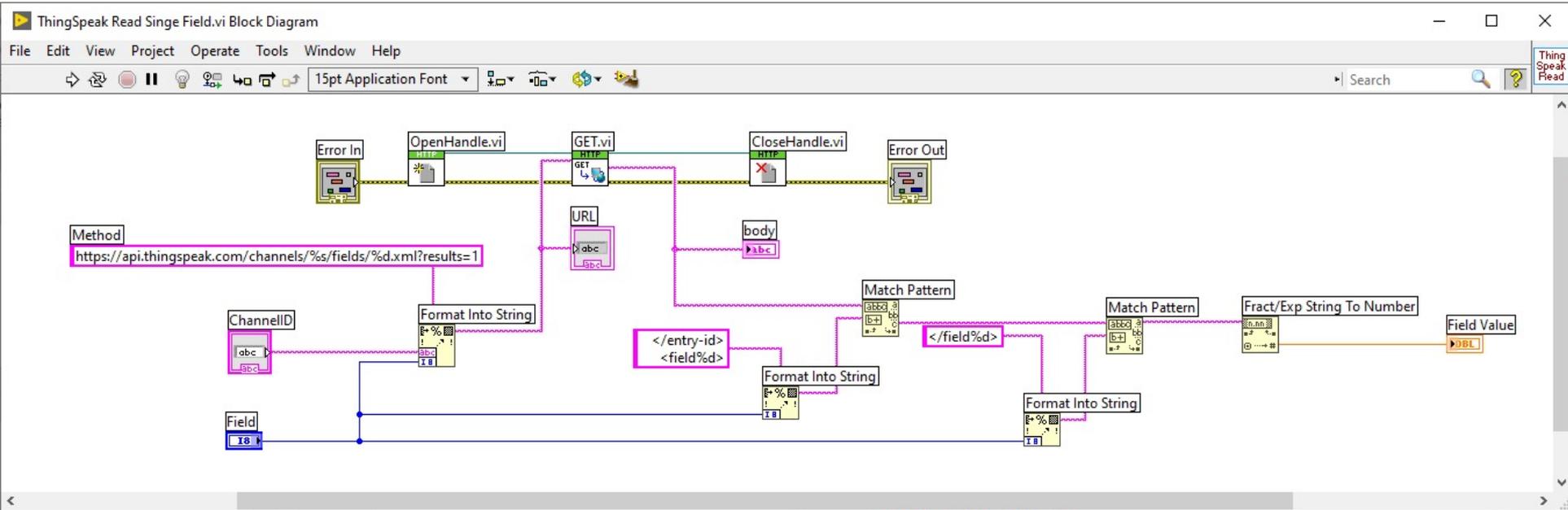
Parsing XML String

In this example, the XML string is parsed in order to find the last updated value for a given Channel and a given Field within that Channel.

You can see the LabVIEW code in detail on the next page.

Block Diagram

Parsing the XML String to get the actual Value

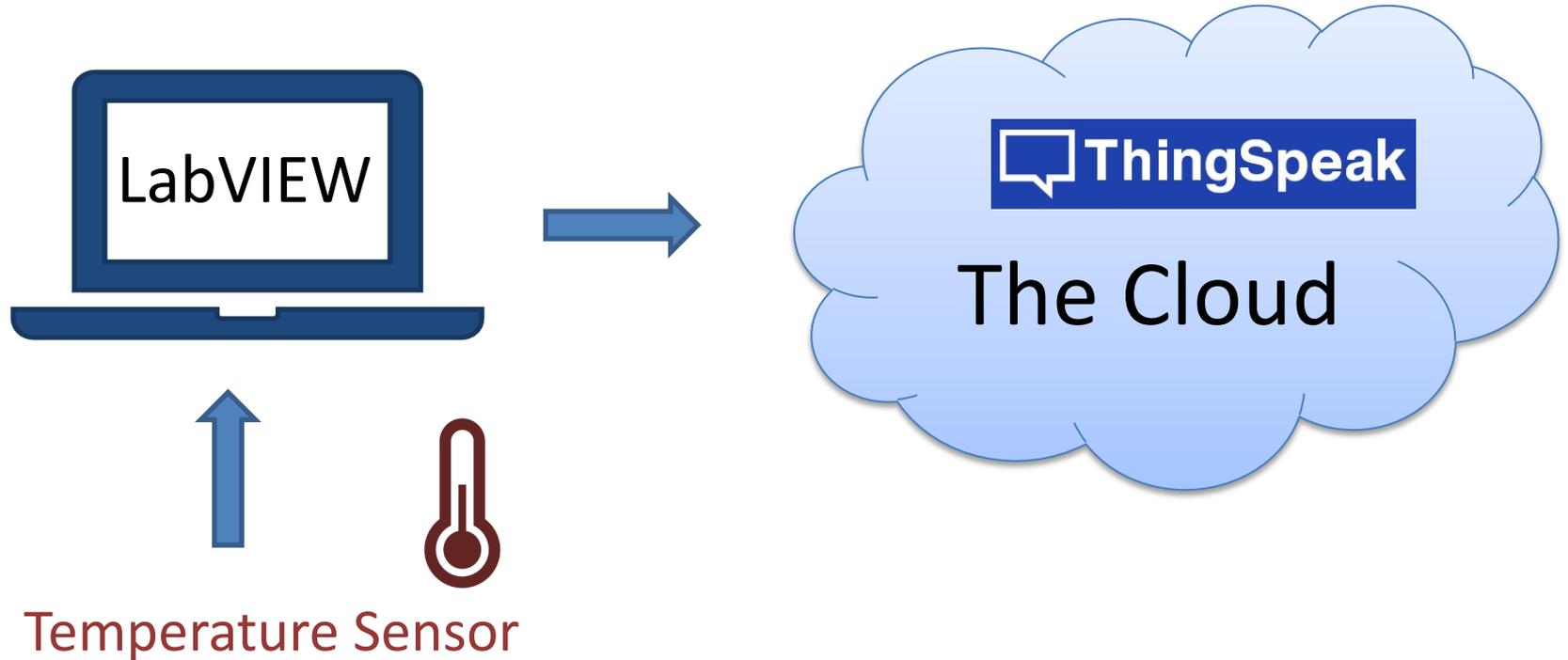




LabVIEW Examples

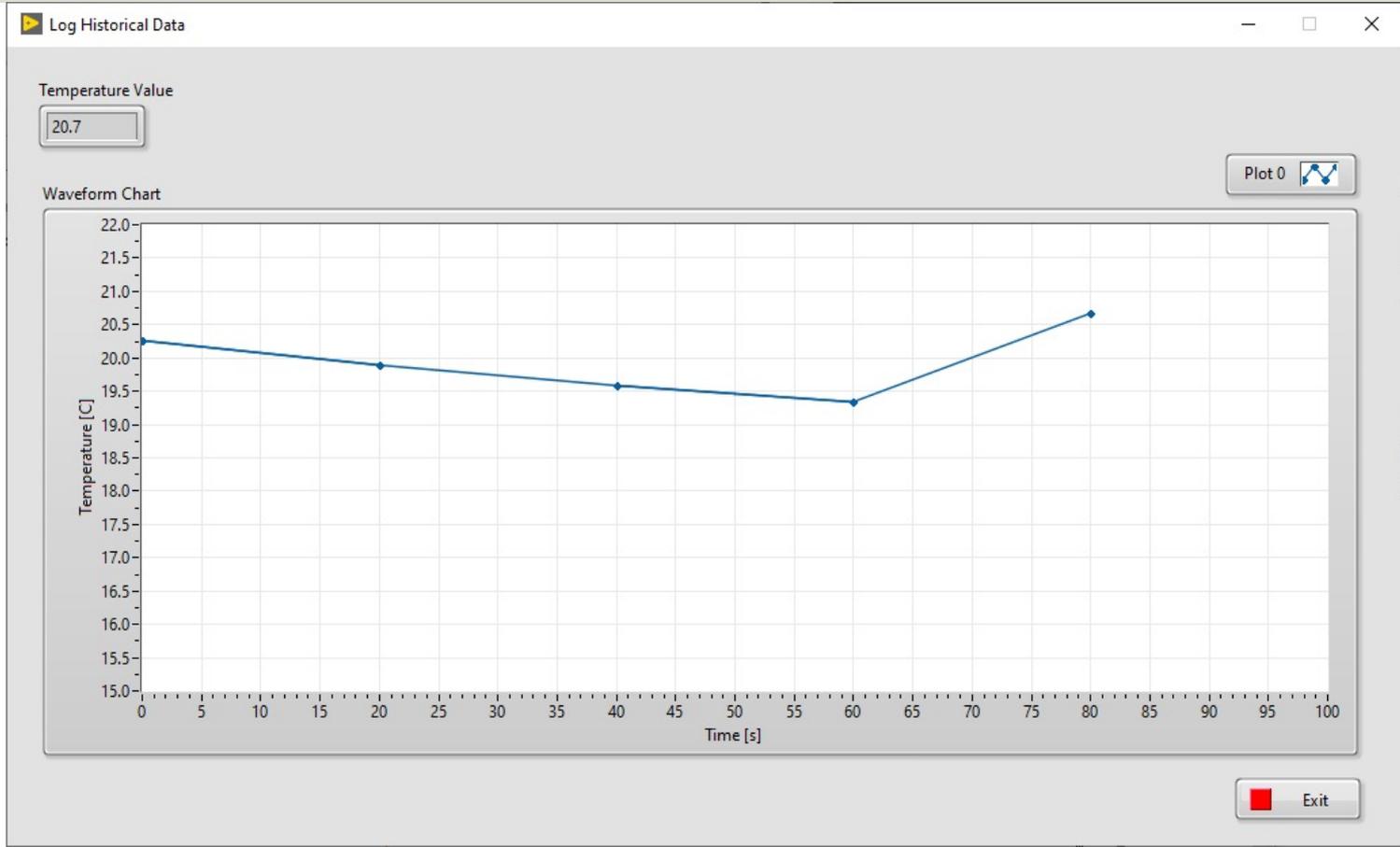
Datalogging

Datalogging Example

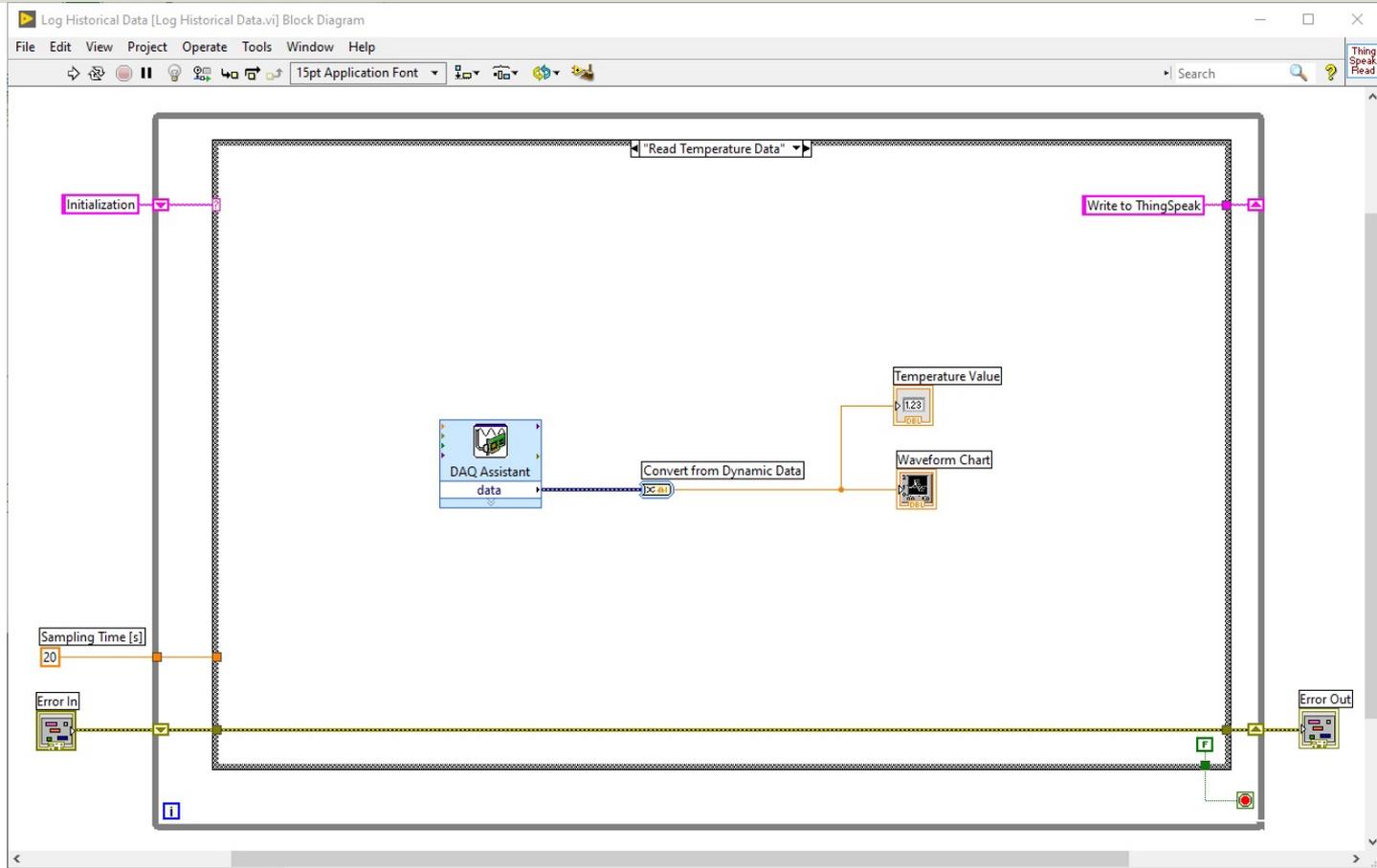


(In this Tutorial a TC-01 Thermocouple Temperature Sensor from National Instruments will be used, but any kind of sensor can be used)

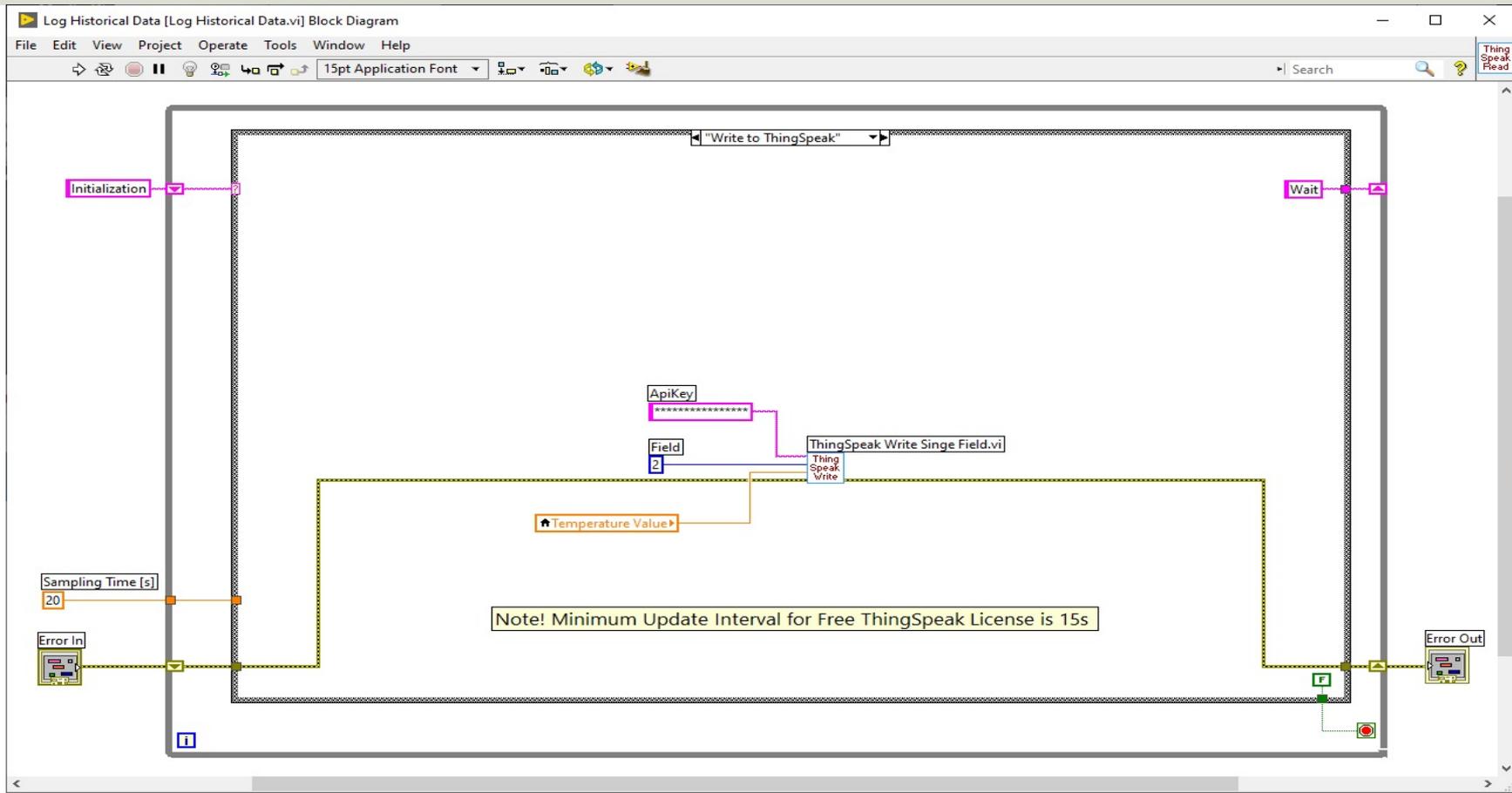
Datalogging Example



Block Diagram – Read Temperature



Block Diagram – Write to ThingSpeak

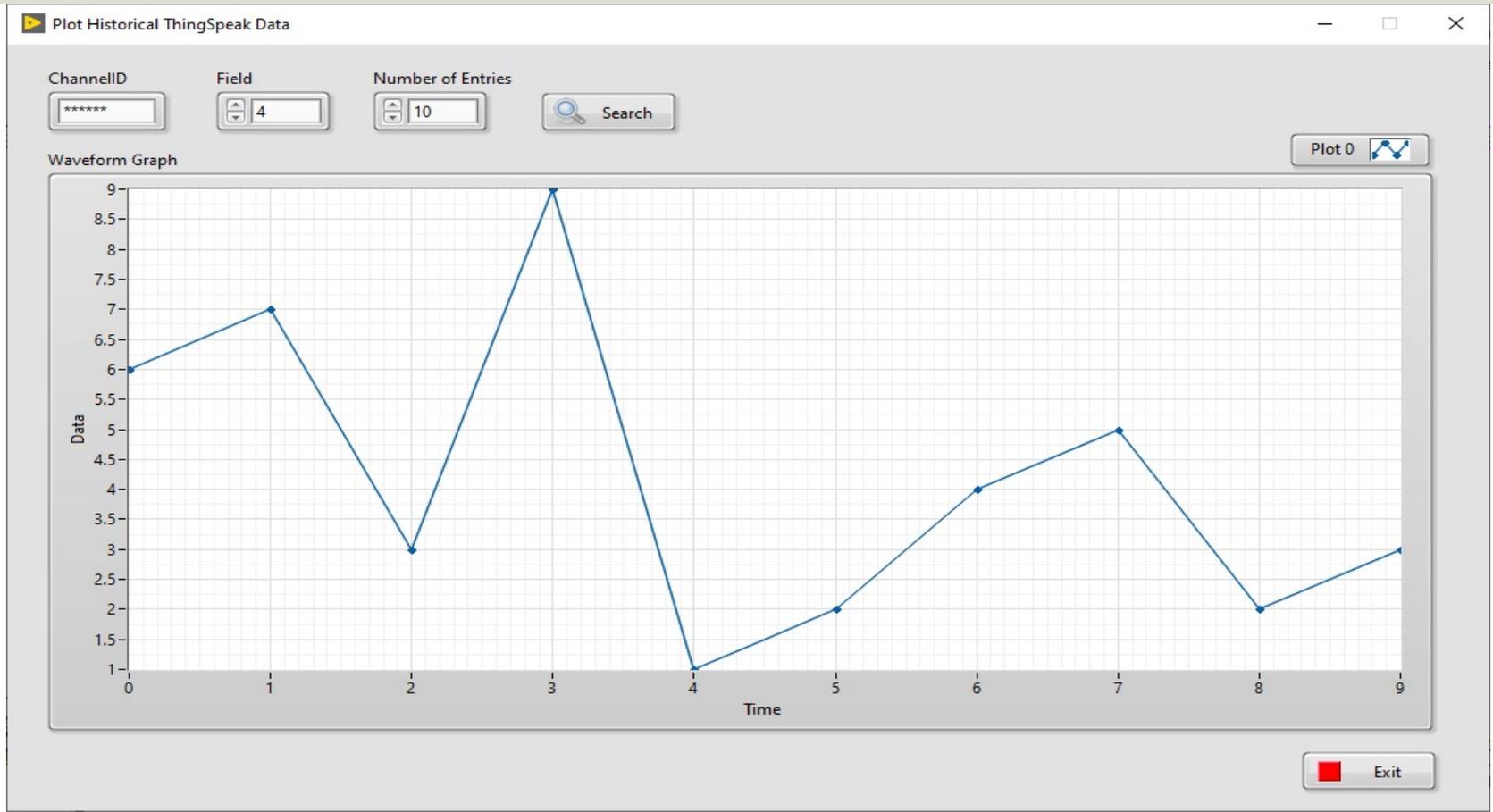




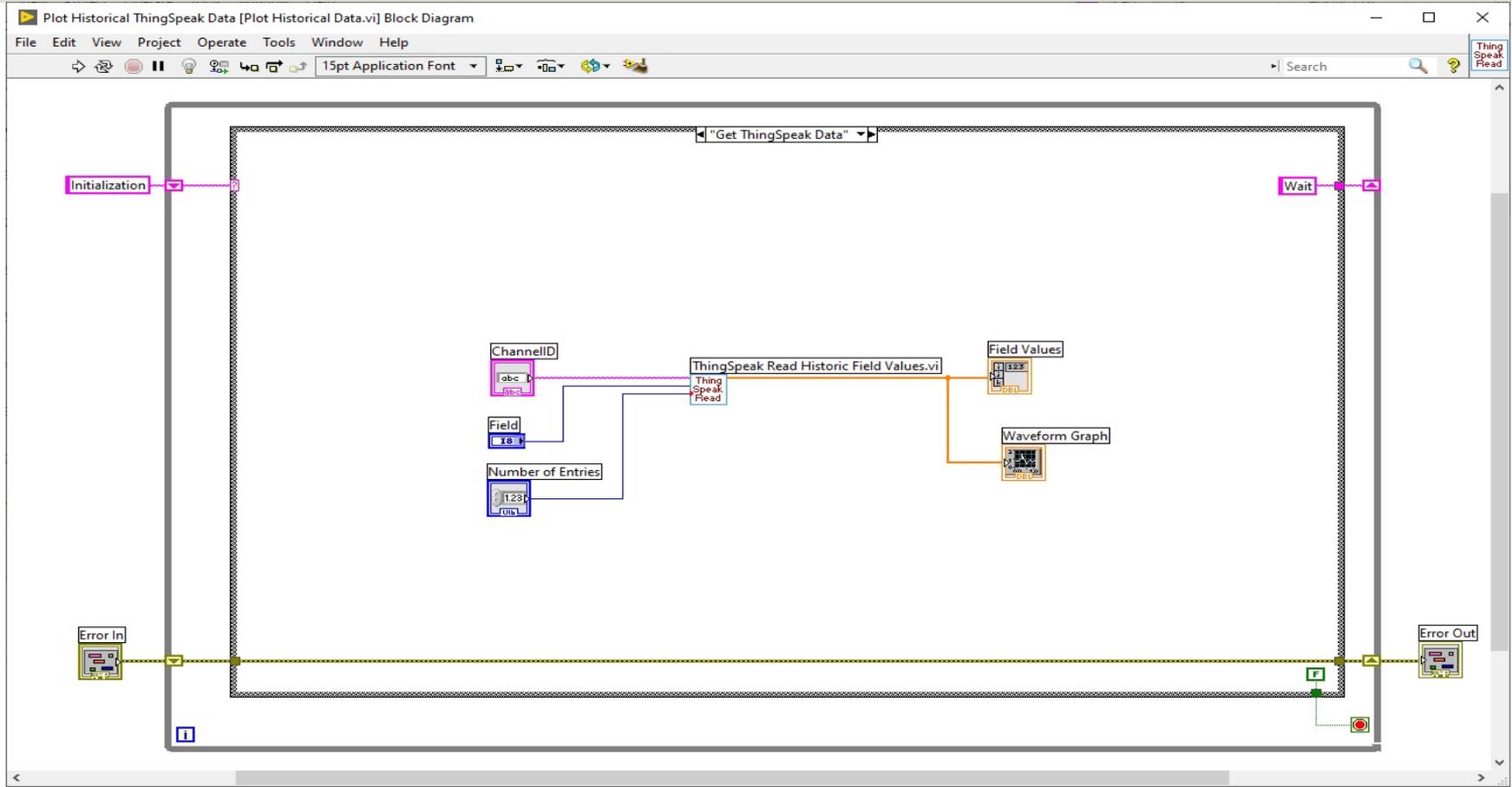
LabVIEW Examples

Read Historical Data from ThingSpeak

Plot Historical Data



Block Diagram



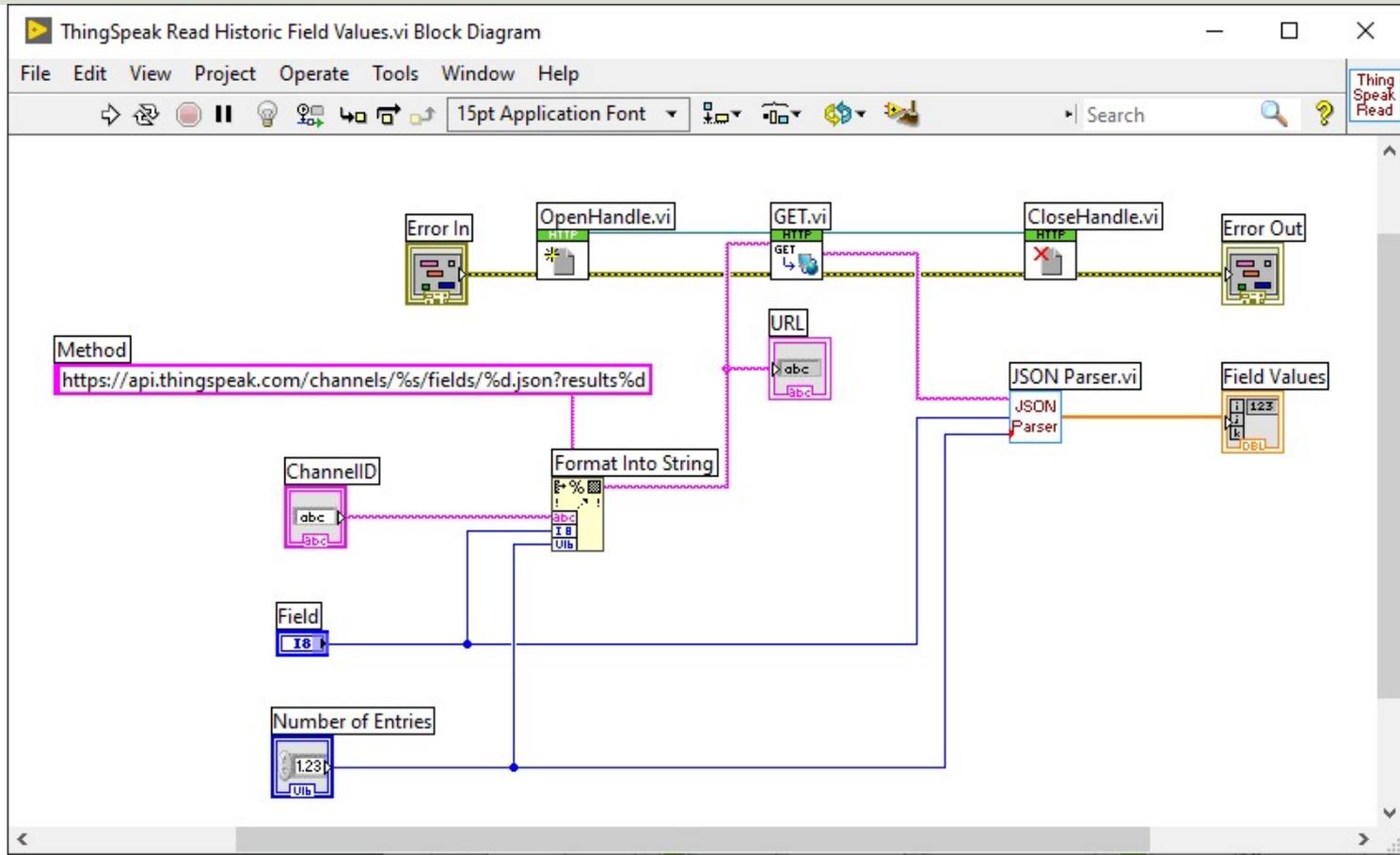
Read Historic Field Values

The screenshot shows the LabVIEW front panel for a VI titled "ThingSpeak Read Historic Field Values.vi". The interface is organized into several sections:

- ChannelID:** A text input field containing "*****".
- Field:** A numeric spinner control set to "4".
- Number of Entries:** A numeric spinner control set to "10".
- URL:** A text input field containing the URL: `https://api.thingspeak.com/channels/ChannelID/fields/4.json?results10`.
- Field Values:** A vertical stack of ten buttons, each representing a field value from 0.00 to 7.00 in increments of 1.00. The values are: 6.00, 7.00, 3.00, 9.00, 1.00, 2.00, 4.00, 5.00, 2.00, 3.00, and 0.00.
- Error In:** A status indicator with a green light and a "code" field set to "0". The "source" field is empty.
- Error Out:** A status indicator with a green light and a "code" field set to "0". The "source" field is empty.

The top of the window shows the LabVIEW menu bar (File, Edit, View, Project, Operate, Tools, Window, Help) and a toolbar with various icons. The font size is set to "15pt Application Font".

Block Diagram



JSON Parser

JSON Parser.vi Front Panel

File Edit View Project Operate Tools Window Help

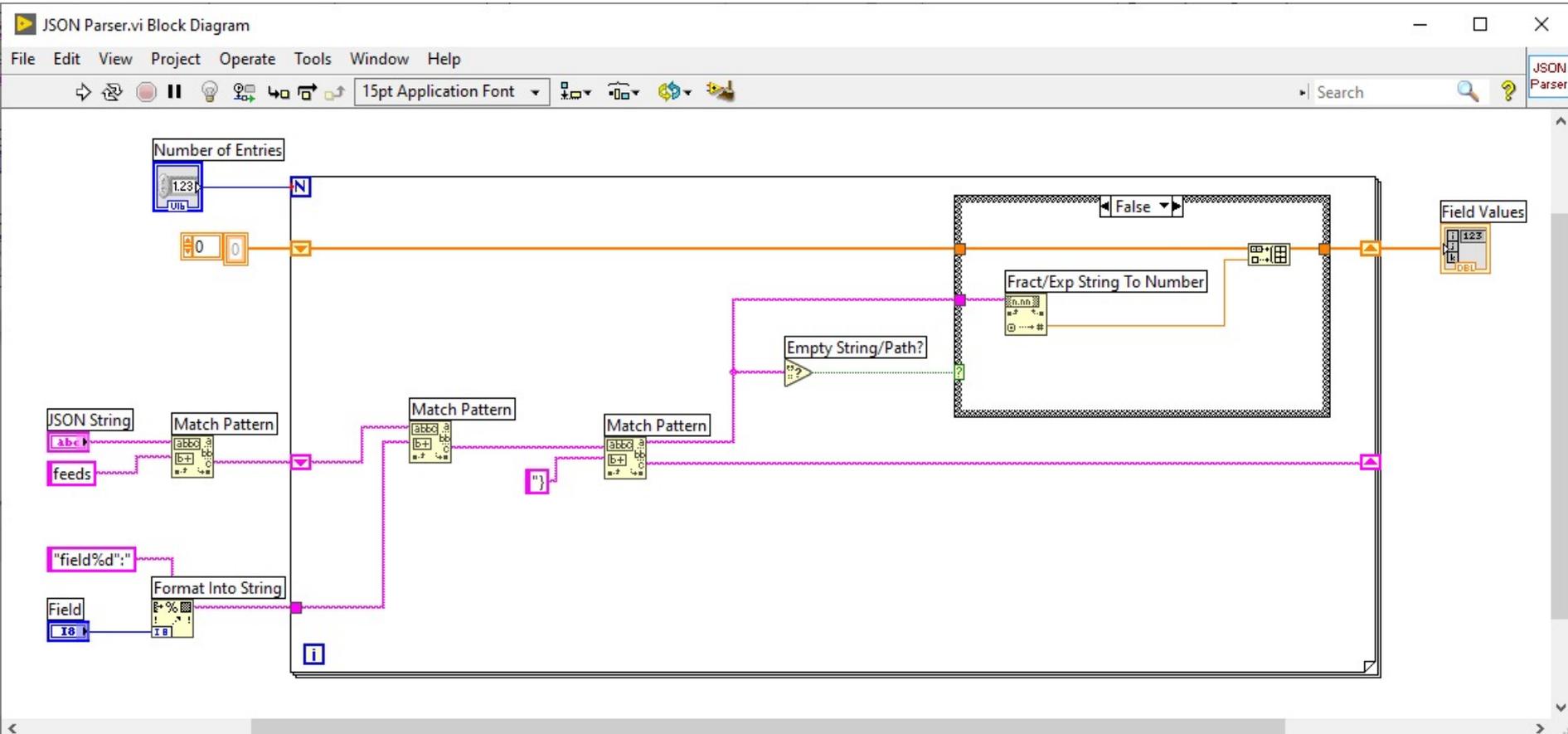
15pt Application Font Search

Field: 4 Number of Entries: 10 Field Values: 0

JSON String

```
{ "channel": { "id": "xxxxxx", "name": "Work", "latitude": "0.0", "longitude": "0.0", "field1": "Office Temperature [C]", "field2": "Temperature B Building [C]", "field3": "Tout", "field4": "Kp", "field5": "Ti", "field6": "SP", "created_at": "2017-05-30T11:41:00Z", "updated_at": "2021-09-08T11:32:07Z", "last_entry_id": 13, "feeds": [ { "created_at": "2021-09-08T12:09:07Z", "entry_id": 1, "field4": "6.00", "created_at": "2021-09-08T12:10:20Z", "entry_id": 2, "field4": "7.00", "created_at": "2021-09-08T12:33Z", "entry_id": 3, "field4": null, "created_at": "2021-09-08T12:13:55Z", "entry_id": 4, "field4": "3.00", "created_at": "2021-09-08T12:15:26Z", "entry_id": 5, "field4": "9.00", "created_at": "2021-09-08T12:22:01Z", "entry_id": 6, "field4": "1.00", "created_at": "2021-09-08T12:22:24Z", "entry_id": 7, "field4": "2.00", "created_at": "2021-09-08T12:24:54Z", "entry_id": 8, "field4": "4.00", "created_at": "2021-09-08T12:26:05Z", "entry_id": 9, "field4": "5.00", "created_at": "2021-09-08T12:48:38Z", "entry_id": 10, "field4": "2.00", "created_at": "2021-09-08T12:49:04Z", "entry_id": 11, "field4": "3.00", "created_at": "2021-09-08T12:54:04Z", "entry_id": 12, "field4": "4.00", "created_at": "2021-09-08T13:03:54Z", "entry_id": 13, "field4": null } ] }
```

Block Diagram



Summary

- ThingSpeak is an IoT service that lets you collect and store sensor data in the cloud and develop Internet of Things applications.
- LabVIEW has powerful features for Simulation, Control and DAQ Applications
- Both ThingSpeak (with limitations, update rate, etc.) and LabVIEW (LabVIEW Community Edition) can be used for free for non-commercial small projects, in addition you can buy a professional license.

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: <https://www.halvorsen.blog>

