Enhancing OCS with Azure Functions

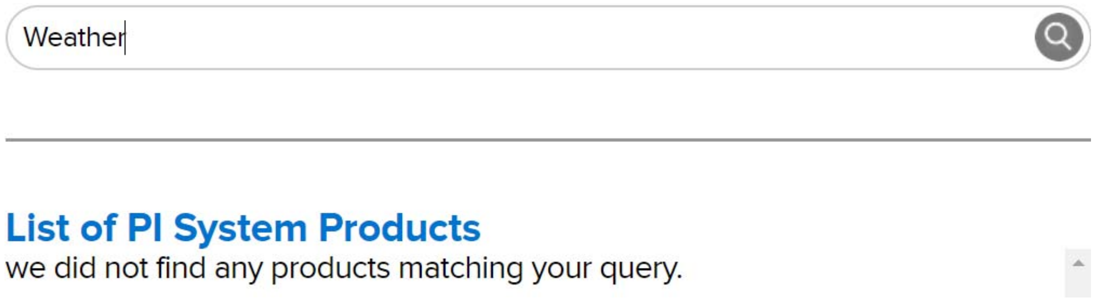
Chad Chisholm

Agenda

- Scenario overview
- Azure Functions overview
- OCS Types and Streams
- Changing the Type of a Stream
- Moving data into OCS via Azure Function
- What can I do with this data in OCS?
- Enhancements for next time

Scenario Overview

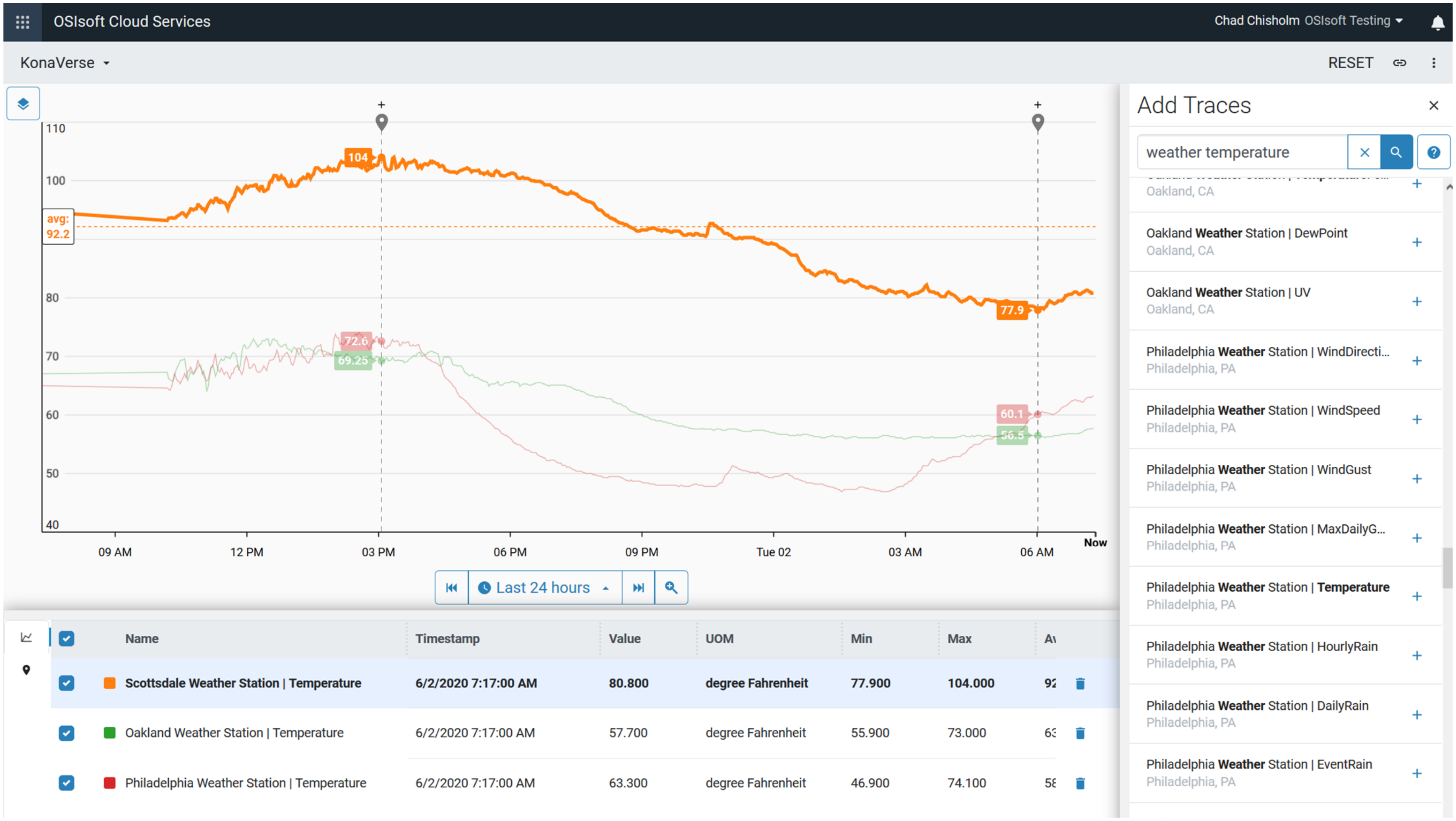
- Software people love hardware
- Weather stations provide real, intuitive data
- I have friends who own weather stations
- I have access to OCS


Weather

List of PI System Products

we did not find any products matching your query.

- Let's build a "Connector"



	Name	Timestamp	Value	UOM	Min	Max	Av
✓ 🟧	**Scottsdale Weather Station | Temperature**	**6/2/2020 7:17:00 AM**	**80.800**	**degree Fahrenheit**	**77.900**	**104.000**	**92**
✓ 🟩	Oakland Weather Station | Temperature	6/2/2020 7:17:00 AM	57.700	degree Fahrenheit	55.900	73.000	63
✓ 🟥	Philadelphia Weather Station | Temperature	6/2/2020 7:17:00 AM	63.300	degree Fahrenheit	46.900	74.100	58

Add Traces

weather temperature

Oakland, CA

Oakland **Weather** Station | DewPoint
Oakland, CA

Oakland **Weather** Station | UV
Oakland, CA

Philadelphia **Weather** Station | WindDirecti...
Philadelphia, PA

Philadelphia **Weather** Station | WindSpeed
Philadelphia, PA

Philadelphia **Weather** Station | WindGust
Philadelphia, PA

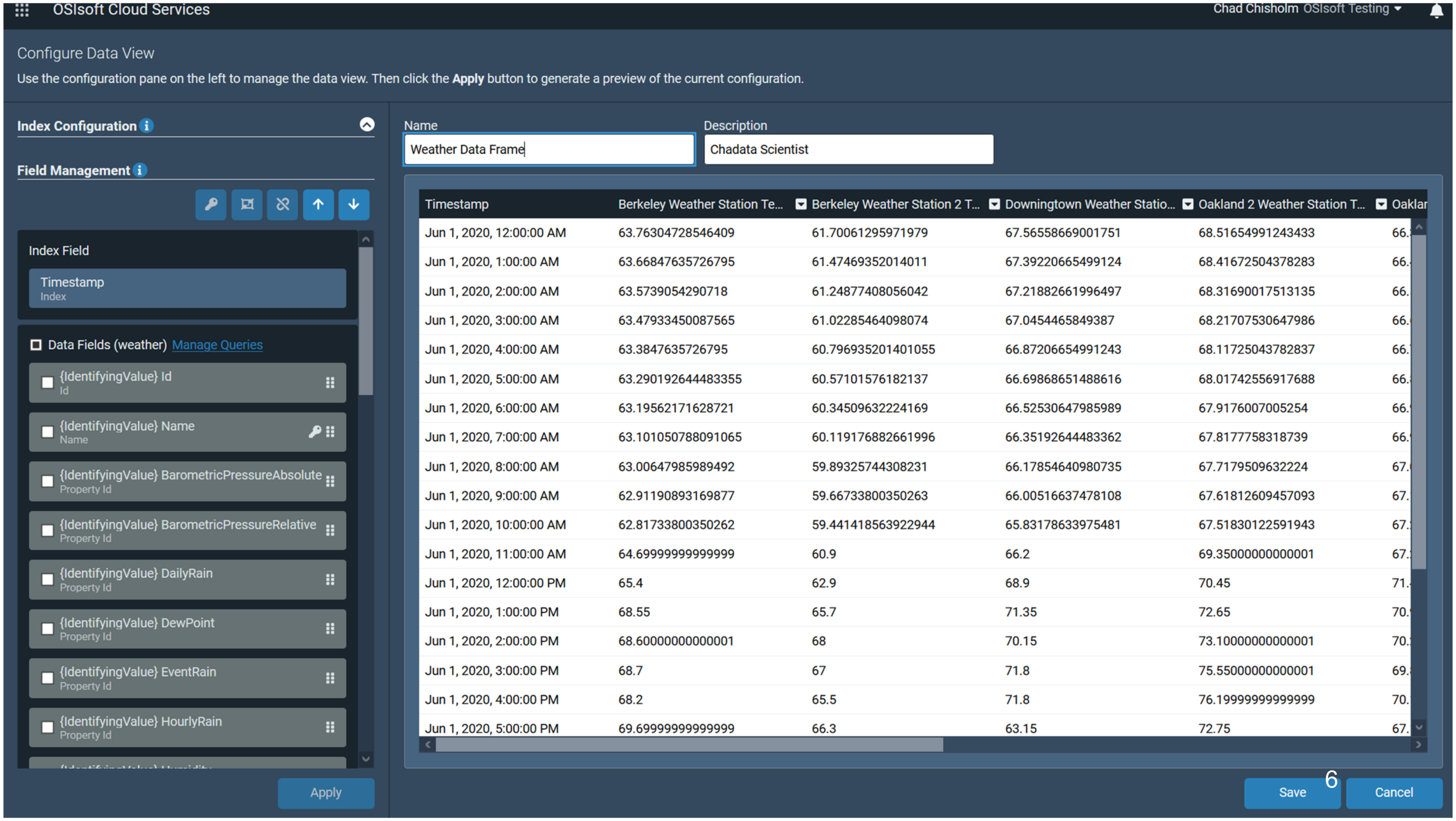
Philadelphia **Weather** Station | MaxDailyG...
Philadelphia, PA

Philadelphia **Weather** Station | **Temperature**
Philadelphia, PA

Philadelphia **Weather** Station | HourlyRain
Philadelphia, PA

Philadelphia **Weather** Station | DailyRain
Philadelphia, PA

Philadelphia **Weather** Station | EventRain
Philadelphia, PA



Configure Data View

Use the configuration pane on the left to manage the data view. Then click the **Apply** button to generate a preview of the current configuration.

Index Configuration

Field Management

Index Field

Timestamp
Index

Data Fields (weather) Manage Queries

- {IdentifyingValue} Id
  Id
- {IdentifyingValue} Name
  Name
- {IdentifyingValue} BarometricPressureAbsolute
  Property Id
- {IdentifyingValue} BarometricPressureRelative
  Property Id
- {IdentifyingValue} DailyRain
  Property Id
- {IdentifyingValue} DewPoint
  Property Id
- {IdentifyingValue} EventRain
  Property Id
- {IdentifyingValue} HourlyRain
  Property Id

Name: Weather Data Frame

Description: Chadata Scientist

Timestamp	Berkeley Weather Station Te...	Berkeley Weather Station 2 T...	Downingtown Weather Statio...	Oakland 2 Weather Station T...	Oaklar
Jun 1, 2020, 12:00:00 AM	63.76304728546409	61.70061295971979	67.56558669001751	68.51654991243433	66.
Jun 1, 2020, 1:00:00 AM	63.66847635726795	61.47469352014011	67.39220665499124	68.41672504378283	66.
Jun 1, 2020, 2:00:00 AM	63.5739054290718	61.24877408056042	67.21882661996497	68.31690017513135	66.
Jun 1, 2020, 3:00:00 AM	63.47933450087565	61.02285464098074	67.0454465849387	68.21707530647986	66.
Jun 1, 2020, 4:00:00 AM	63.3847635726795	60.796935201401055	66.87206654991243	68.11725043782837	66.
Jun 1, 2020, 5:00:00 AM	63.290192644483355	60.57101576182137	66.69868651488616	68.01742556917688	66.
Jun 1, 2020, 6:00:00 AM	63.19562171628721	60.34509632224169	66.52530647985989	67.9176007005254	66.
Jun 1, 2020, 7:00:00 AM	63.101050788091065	60.119176882661996	66.35192644483362	67.8177758318739	66.
Jun 1, 2020, 8:00:00 AM	63.00647985989492	59.89325744308231	66.17854640980735	67.7179509632224	67.
Jun 1, 2020, 9:00:00 AM	62.91190893169877	59.66733800350263	66.00516637478108	67.61812609457093	67.
Jun 1, 2020, 10:00:00 AM	62.81733800350262	59.441418563922944	65.83178633975481	67.51830122591943	67.
Jun 1, 2020, 11:00:00 AM	64.69999999999999	60.9	66.2	69.35000000000001	67.
Jun 1, 2020, 12:00:00 PM	65.4	62.9	68.9	70.45	71.
Jun 1, 2020, 1:00:00 PM	68.55	65.7	71.35	72.65	70.
Jun 1, 2020, 2:00:00 PM	68.60000000000001	68	70.15	73.10000000000001	70.
Jun 1, 2020, 3:00:00 PM	68.7	67	71.8	75.55000000000001	69.
Jun 1, 2020, 4:00:00 PM	68.2	65.5	71.8	76.19999999999999	70.
Jun 1, 2020, 5:00:00 PM	69.69999999999999	66.3	63.15	72.75	67.

6

Azure Functions - Overview

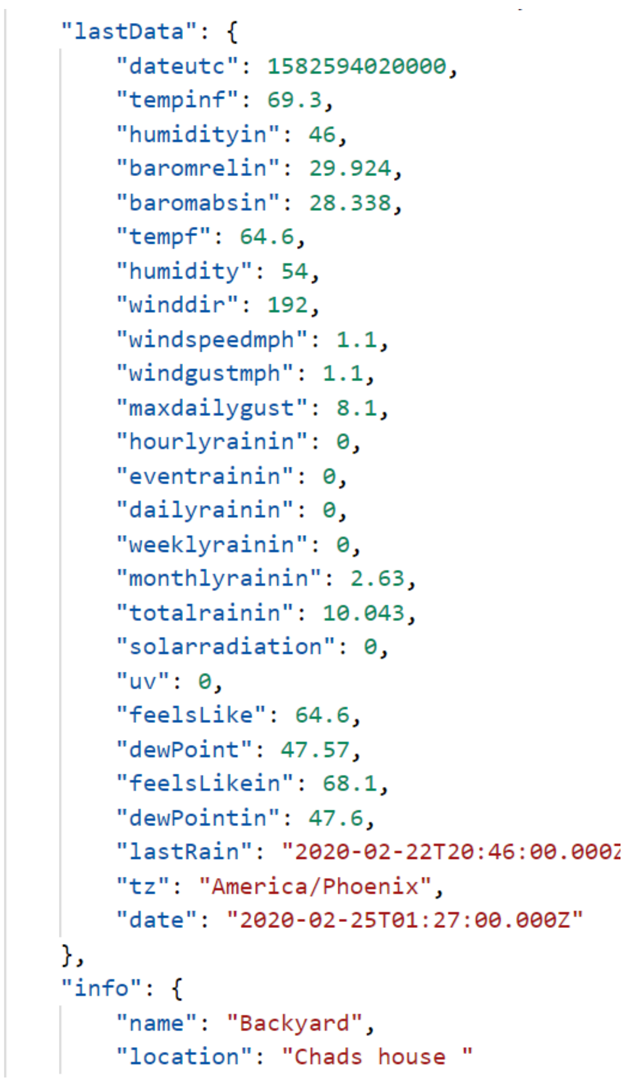
- **Serverless** Applications
  - I’m certain there are servers somewhere
- Use **any language** you want!
  - `As long as it’s C#, Java, JavaScript, Python, or PowerShell`

Azure Functions - Overview

- Trigger your function by **Timer**
  - That’s what we’re doing today
- Trigger your function from **Azure Something**
  - Blob changes, Queue messages
  - Event Hub events (single or batch)
  - Service Bus topics or queues
  - Webhook – triggered by an Http request

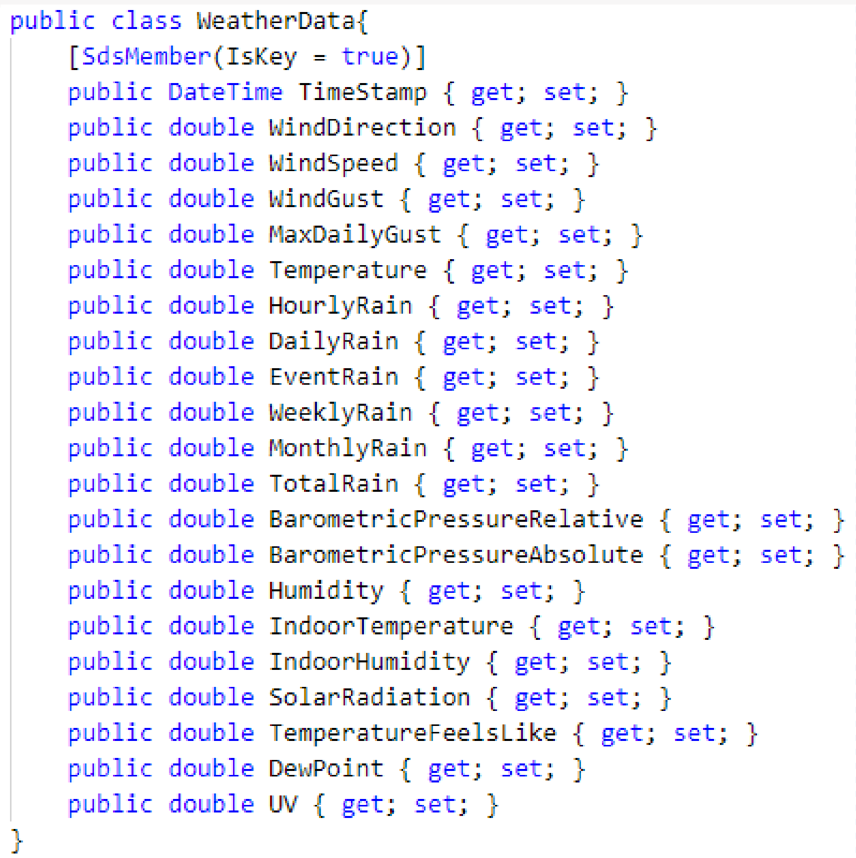
Vendor data API

```
GET https:///api….net/v1...
?apiKey=7ee5…
&applicationKey=debe...
```



```
"lastData": {
    "dateutc": 1582594020000,
    "tempinf": 69.3,
    "humidityin": 46,
    "baromrelin": 29.924,
    "baromabsin": 28.338,
    "tempf": 64.6,
    "humidity": 54,
    "winddir": 192,
    "windspeedmph": 1.1,
    "windgustmph": 1.1,
    "maxdailygust": 8.1,
    "hourlyrainin": 0,
    "eventrainin": 0,
    "dailyrainin": 0,
    "weeklyrainin": 0,
    "monthlyrainin": 2.63,
    "totalrainin": 10.043,
    "solarradiation": 0,
    "uv": 0,
    "feelsLike": 64.6,
    "dewPoint": 47.57,
    "feelsLikein": 68.1,
    "dewPointin": 47.6,
    "lastRain": "2020-02-22T20:46:00.000Z
    "tz": "America/Phoenix",
    "date": "2020-02-25T01:27:00.000Z"
},
"info": {
    "name": "Backyard",
    "location": "Chads house "
```

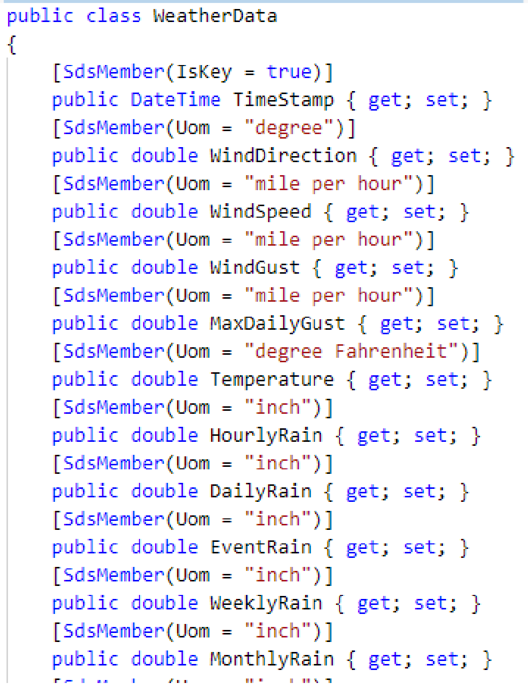
Designing OCS Types and Streams



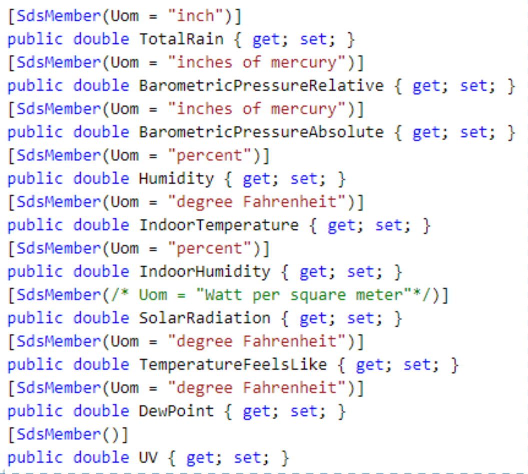
```
public class WeatherData{
    [SdsMember(IsKey = true)]
    public DateTime TimeStamp { get; set; }
    public double WindDirection { get; set; }
    public double WindSpeed { get; set; }
    public double WindGust { get; set; }
    public double MaxDailyGust { get; set; }
    public double Temperature { get; set; }
    public double HourlyRain { get; set; }
    public double DailyRain { get; set; }
    public double EventRain { get; set; }
    public double WeeklyRain { get; set; }
    public double MonthlyRain { get; set; }
    public double TotalRain { get; set; }
    public double BarometricPressureRelative { get; set; }
    public double BarometricPressureAbsolute { get; set; }
    public double Humidity { get; set; }
    public double IndoorTemperature { get; set; }
    public double IndoorHumidity { get; set; }
    public double SolarRadiation { get; set; }
    public double TemperatureFeelsLike { get; set; }
    public double DewPoint { get; set; }
    public double UV { get; set; }
}
```

Changing the Stream Type

I forgot UoMs!



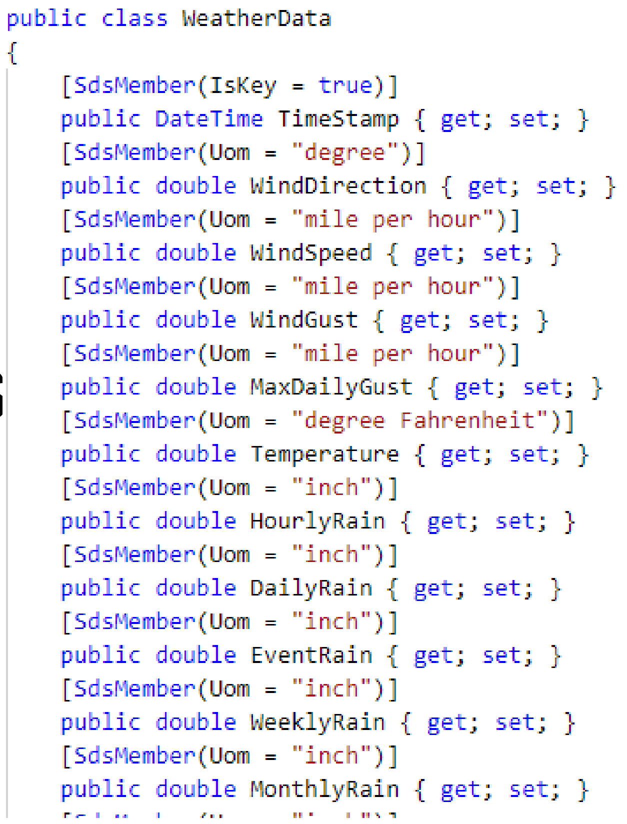
```
public class WeatherData
{
    [SdsMember(IsKey = true)]
    public DateTime TimeStamp { get; set; }
    [SdsMember(Uom = "degree")]
    public double WindDirection { get; set; }
    [SdsMember(Uom = "mile per hour")]
    public double WindSpeed { get; set; }
    [SdsMember(Uom = "mile per hour")]
    public double WindGust { get; set; }
    [SdsMember(Uom = "mile per hour")]
    public double MaxDailyGust { get; set; }
    [SdsMember(Uom = "degree Fahrenheit")]
    public double Temperature { get; set; }
    [SdsMember(Uom = "inch")]
    public double HourlyRain { get; set; }
    [SdsMember(Uom = "inch")]
    public double DailyRain { get; set; }
    [SdsMember(Uom = "inch")]
    public double EventRain { get; set; }
    [SdsMember(Uom = "inch")]
    public double WeeklyRain { get; set; }
    [SdsMember(Uom = "inch")]
    public double MonthlyRain { get; set; }
```



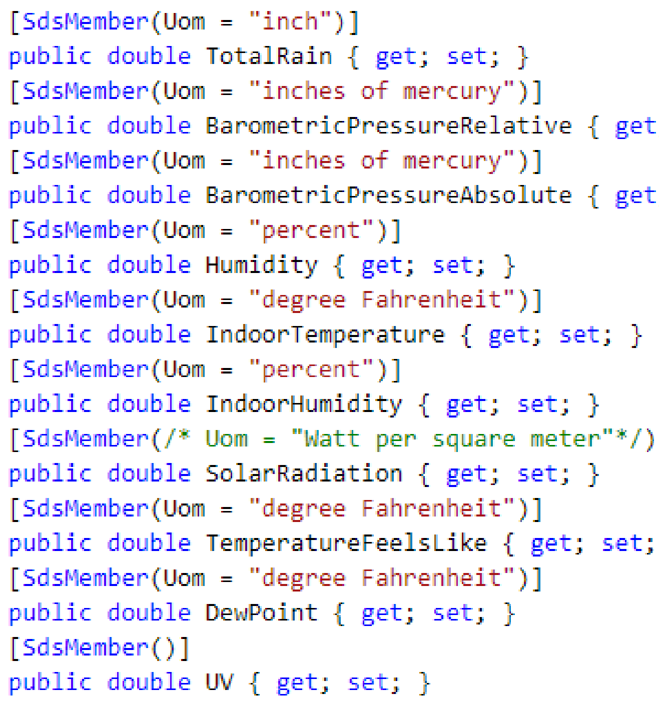
```
[SdsMember(Uom = "inch")]
public double TotalRain { get; set; }
[SdsMember(Uom = "inches of mercury")]
public double BarometricPressureRelative { get; set; }
[SdsMember(Uom = "inches of mercury")]
public double BarometricPressureAbsolute { get; set; }
[SdsMember(Uom = "percent")]
public double Humidity { get; set; }
[SdsMember(Uom = "degree Fahrenheit")]
public double IndoorTemperature { get; set; }
[SdsMember(Uom = "percent")]
public double IndoorHumidity { get; set; }
[SdsMember(/* Uom = "Watt per square meter"*/)]
public double SolarRadiation { get; set; }
[SdsMember(Uom = "degree Fahrenheit")]
public double TemperatureFeelsLike { get; set; }
[SdsMember(Uom = "degree Fahrenheit")]
public double DewPoint { get; set; }
[SdsMember()]
public double UV { get; set; }
```

Designing OCS Types and Streams

Don’t forgot UoMs!



```
public class WeatherData
{
    [SdsMember(IsKey = true)]
    public DateTime TimeStamp { get; set; }
    [SdsMember(Uom = "degree")]
    public double WindDirection { get; set; }
    [SdsMember(Uom = "mile per hour")]
    public double WindSpeed { get; set; }
    [SdsMember(Uom = "mile per hour")]
    public double WindGust { get; set; }
    [SdsMember(Uom = "mile per hour")]
    public double MaxDailyGust { get; set; }
    [SdsMember(Uom = "degree Fahrenheit")]
    public double Temperature { get; set; }
    [SdsMember(Uom = "inch")]
    public double HourlyRain { get; set; }
    [SdsMember(Uom = "inch")]
    public double DailyRain { get; set; }
    [SdsMember(Uom = "inch")]
    public double EventRain { get; set; }
    [SdsMember(Uom = "inch")]
    public double WeeklyRain { get; set; }
    [SdsMember(Uom = "inch")]
    public double MonthlyRain { get; set; }
```



```
[SdsMember(Uom = "inch")]
public double TotalRain { get; set; }
[SdsMember(Uom = "inches of mercury")]
public double BarometricPressureRelative { get
[SdsMember(Uom = "inches of mercury")]
public double BarometricPressureAbsolute { get
[SdsMember(Uom = "percent")]
public double Humidity { get; set; }
[SdsMember(Uom = "degree Fahrenheit")]
public double IndoorTemperature { get; set; }
[SdsMember(Uom = "percent")]
public double IndoorHumidity { get; set; }
[SdsMember(/* Uom = "Watt per square meter"*/)
public double SolarRadiation { get; set; }
[SdsMember(Uom = "degree Fahrenheit")]
public double TemperatureFeelsLike { get; set;
[SdsMember(Uom = "degree Fahrenheit")]
public double DewPoint { get; set; }
[SdsMember()]
public double UV { get; set; }
```

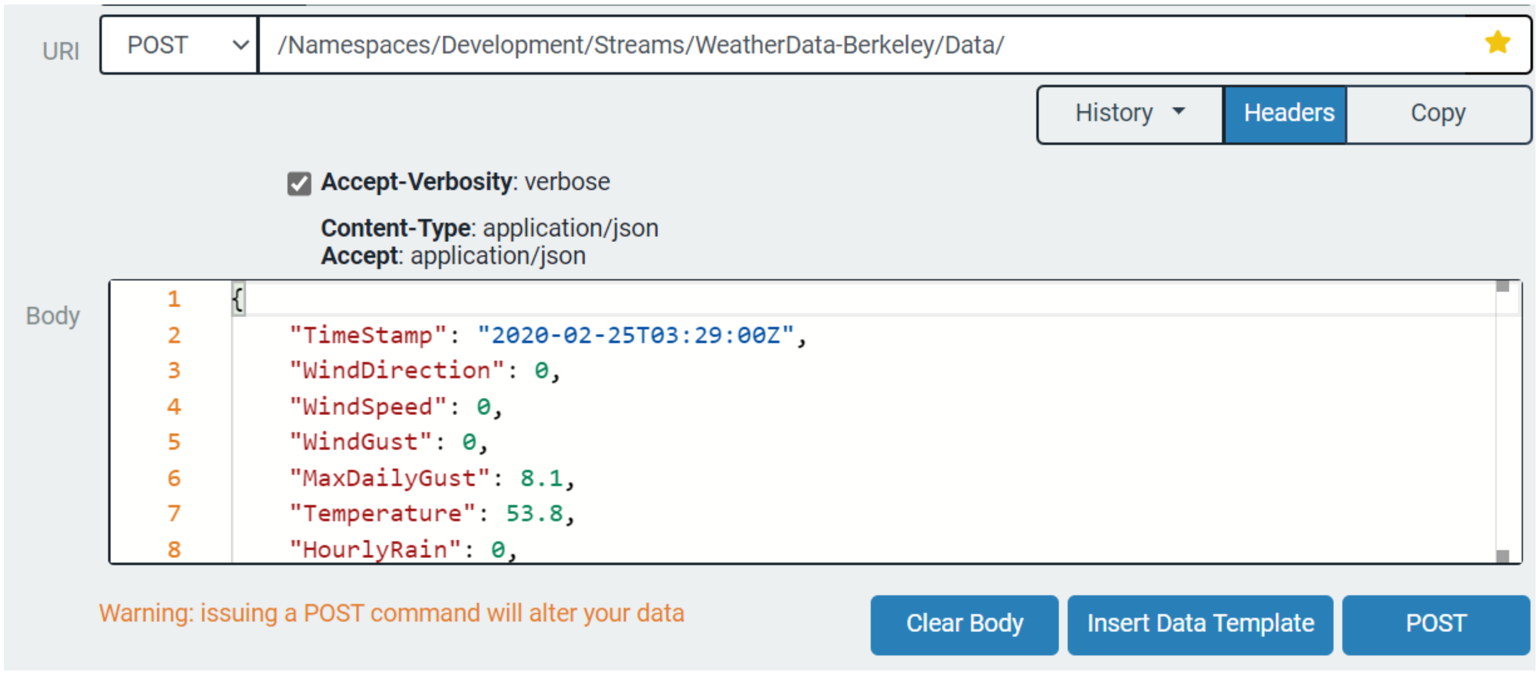
Designing OCS Types and Streams

- C# type → SDS Type

```
SdsType newtype = SdsTypeBuilder.CreateSdsType<WeatherData>();
newtype.Id = typeId;
metadata.GetOrCreateTypeAsync(newtype).GetAwaiter().GetResult();
```

Moving Data to OCS via Azure Functions

- via direct HTTP API



URI POST /Namespaces/Development/Streams/WeatherData-Berkeley/Data/

☑ **Accept-Verbosity**: verbose

Content-Type: application/json
Accept: application/json

Body

```
{
    "TimeStamp": "2020-02-25T03:29:00Z",
    "WindDirection": 0,
    "WindSpeed": 0,
    "WindGust": 0,
    "MaxDailyGust": 8.1,
    "Temperature": 53.8,
    "HourlyRain": 0,
```

Warning: issuing a POST command will alter your data

Moving Data to OCS via Azure Functions

- via direct HTTP API

```
response = await httpClient.PostAsync("https://dat-b.osisoft.com/api/v1/tenants/..
```

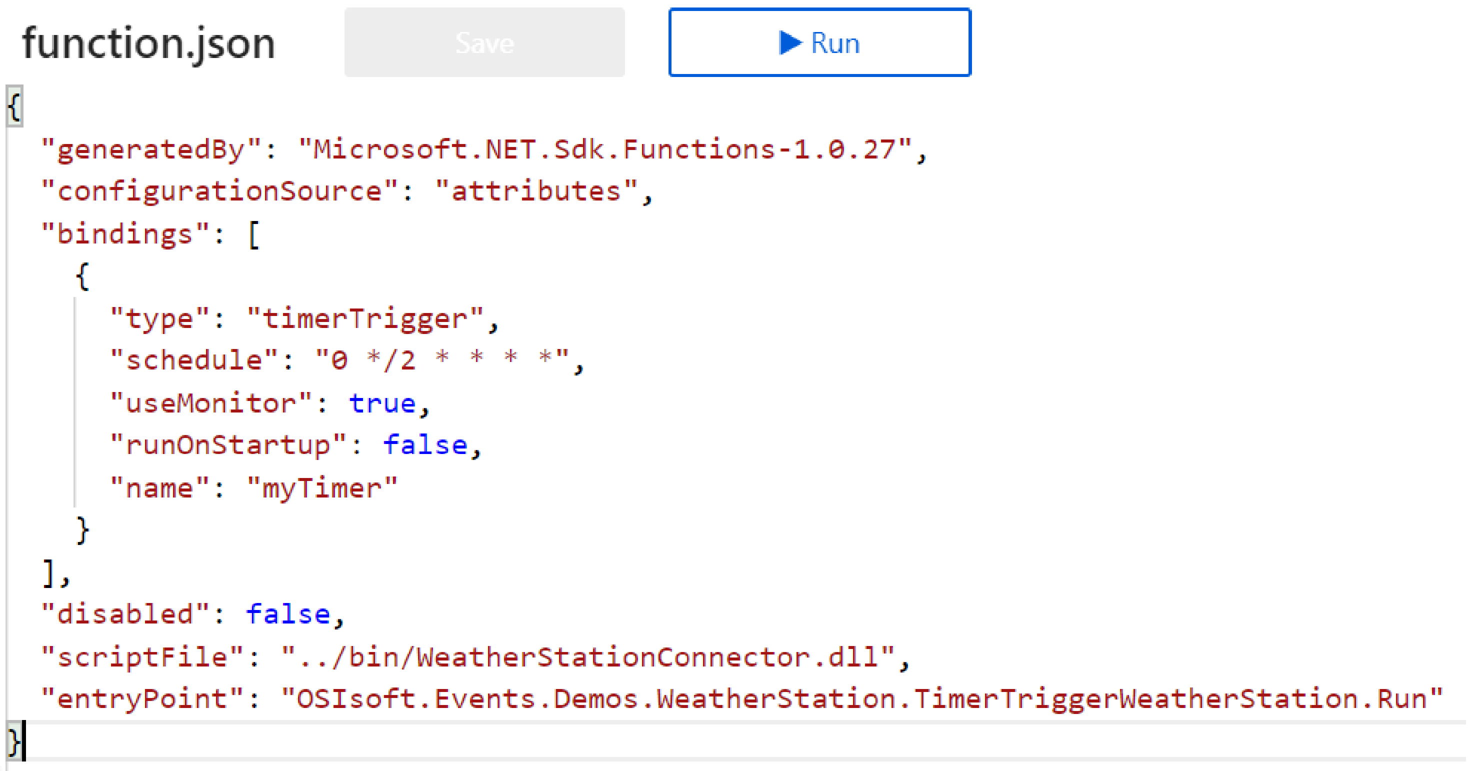
Moving Data to OCS via Azure Functions

- via OCS Client Libraries

```
> dotnet add package OSIsoft.OCSClients
```

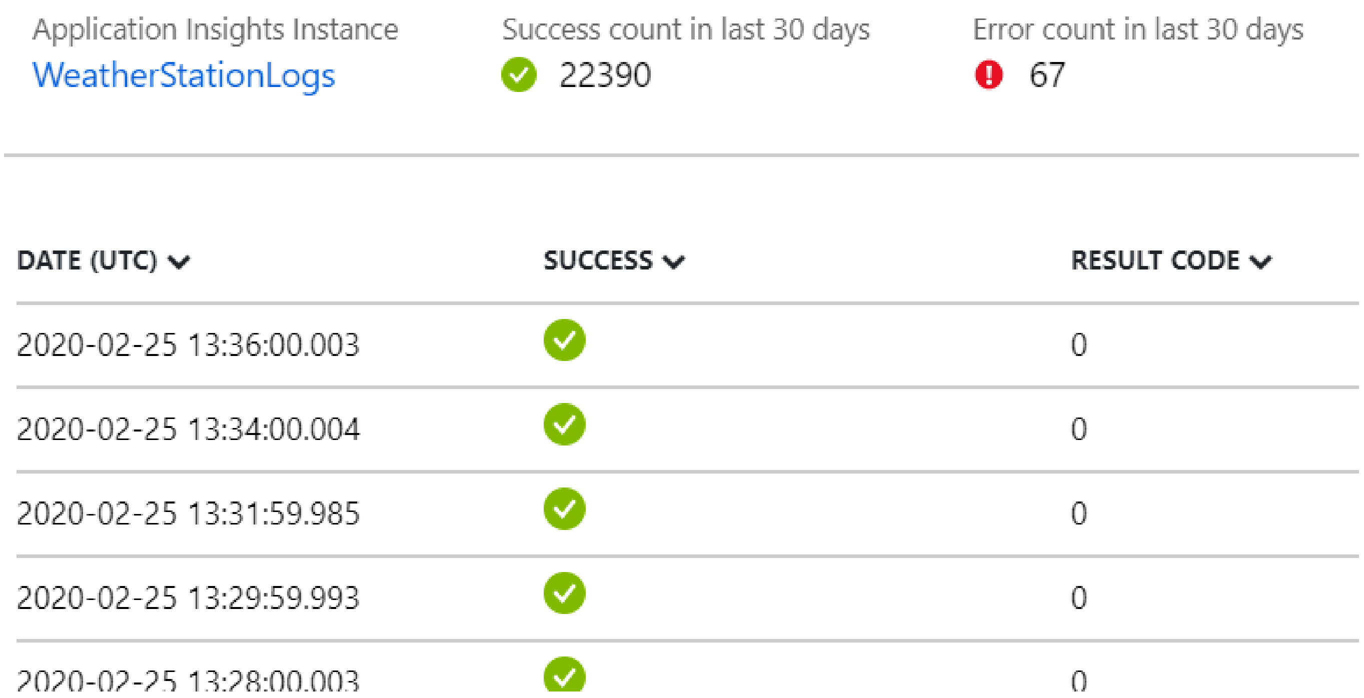
```
ocs.DataService.InsertValueAsync<WeatherData>(connection.StreamId, connection.WeatherData).
```

Azure Function – Startup Config



```
{
  "generatedBy": "Microsoft.NET.Sdk.Functions-1.0.27",
  "configurationSource": "attributes",
  "bindings": [
    {
      "type": "timerTrigger",
      "schedule": "0 */2 * * * *",
      "useMonitor": true,
      "runOnStartup": false,
      "name": "myTimer"
    }
  ],
  "disabled": false,
  "scriptFile": "../bin/WeatherStationConnector.dll",
  "entryPoint": "OSIsoft.Events.Demos.WeatherStation.TimerTriggerWeatherStation.Run"
}
```

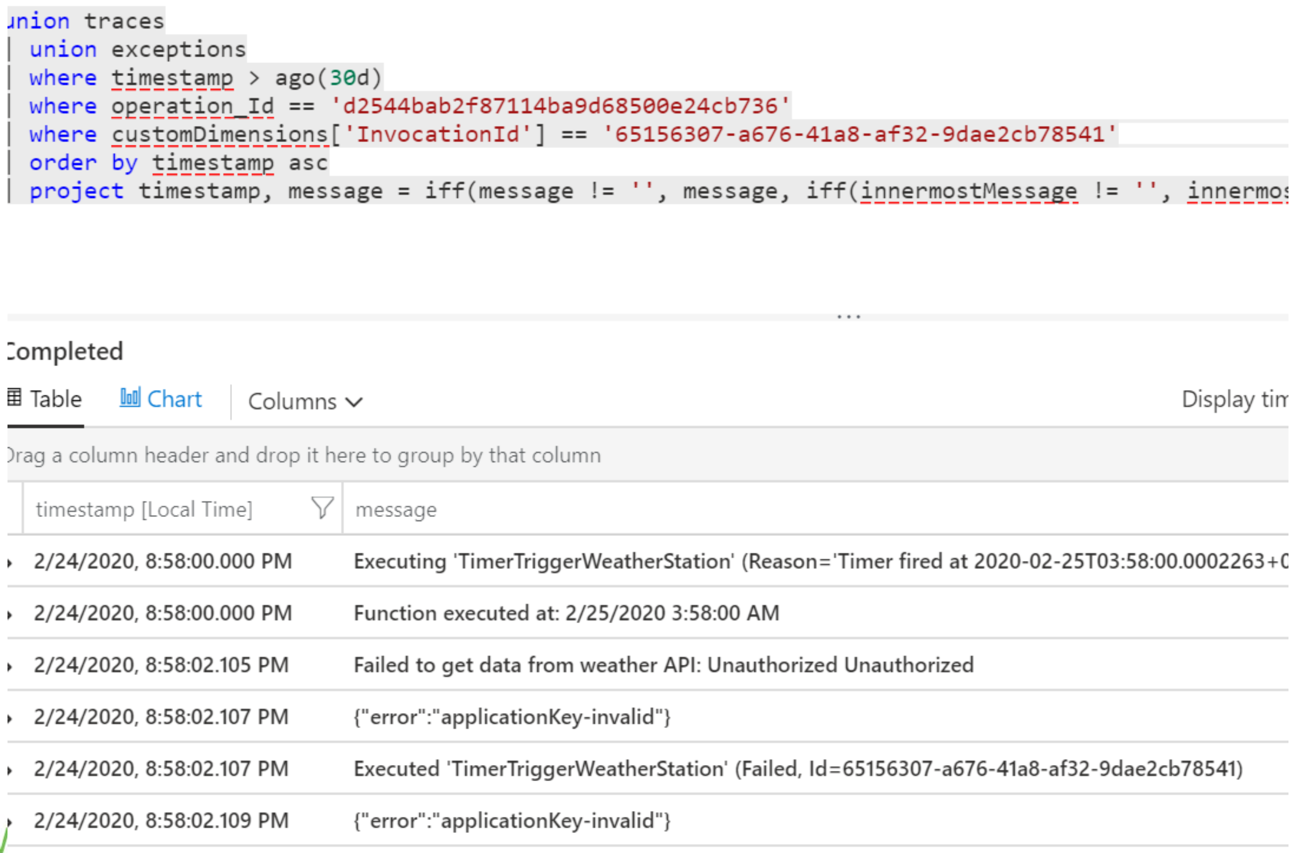
Azure Function – Monitoring



Application Insights Instance	Success count in last 30 days	Error count in last 30 days
WeatherStationLogs	✓ 22390	! 67

DATE (UTC)	SUCCESS	RESULT CODE
2020-02-25 13:36:00.003	✓	0
2020-02-25 13:34:00.004	✓	0
2020-02-25 13:31:59.985	✓	0
2020-02-25 13:29:59.993	✓	0
2020-02-25 13:28:00.003	✓	0

Azure Function – Monitoring



```
union traces
| union exceptions
| where timestamp > ago(30d)
| where operation_Id == 'd2544bab2f87114ba9d68500e24cb736'
| where customDimensions['InvocationId'] == '65156307-a676-41a8-af32-9dae2cb78541'
| order by timestamp asc
| project timestamp, message = iff(message != '', message, iff(innermostMessage != '', innermos
```

Completed

Table | Chart | Columns | Display tim

Drag a column header and drop it here to group by that column

timestamp [Local Time]	message
2/24/2020, 8:58:00.000 PM	Executing 'TimerTriggerWeatherStation' (Reason='Timer fired at 2020-02-25T03:58:00.0002263+0
2/24/2020, 8:58:00.000 PM	Function executed at: 2/25/2020 3:58:00 AM
2/24/2020, 8:58:02.105 PM	Failed to get data from weather API: Unauthorized Unauthorized
2/24/2020, 8:58:02.107 PM	{"error":"applicationKey-invalid"}
2/24/2020, 8:58:02.107 PM	Executed 'TimerTriggerWeatherStation' (Failed, Id=65156307-a676-41a8-af32-9dae2cb78541)
2/24/2020, 8:58:02.109 PM	{"error":"applicationKey-invalid"}

We need more data!

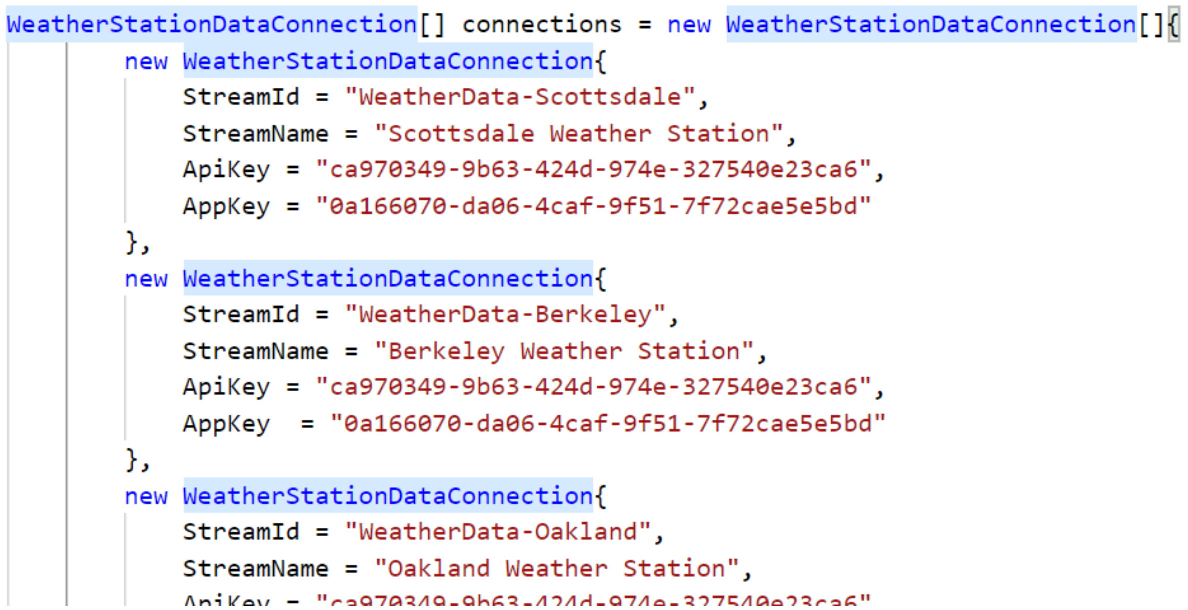
- Despite what they say, I have friends

OSIsoft.

Maui Sodas

We need more data! - Options

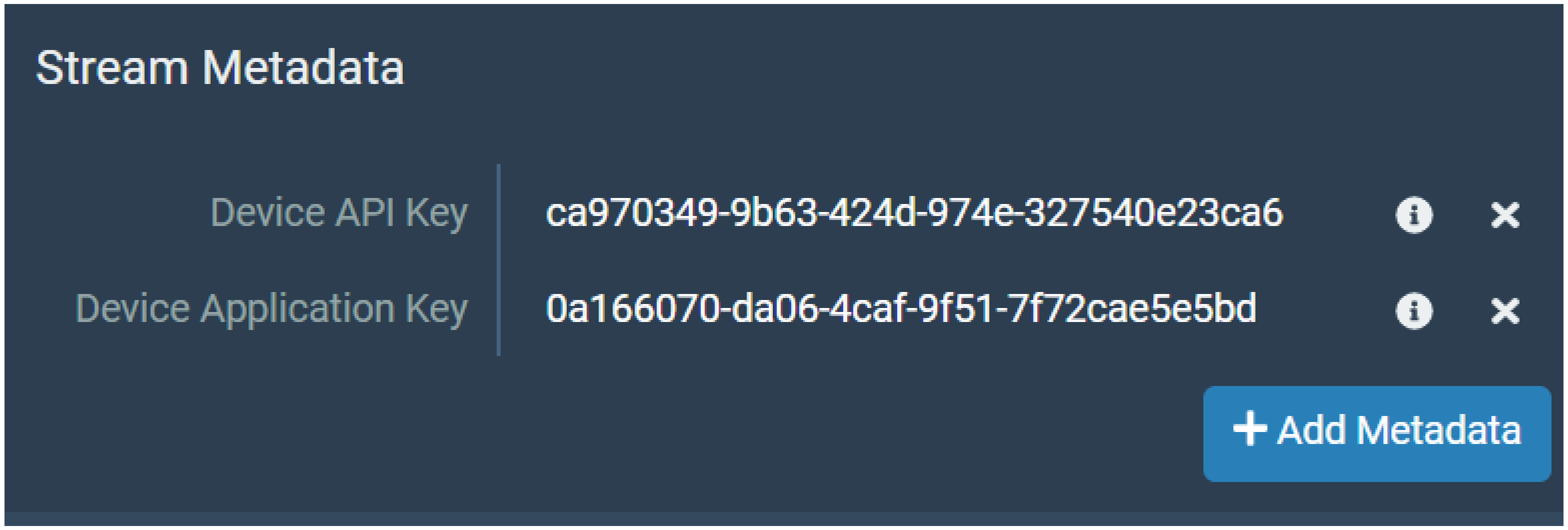
- Through the cunning use of FOR LOOPS!



```
WeatherStationDataConnection[] connections = new WeatherStationDataConnection[]{
        new WeatherStationDataConnection{
            StreamId = "WeatherData-Scottsdale",
            StreamName = "Scottsdale Weather Station",
            ApiKey = "ca970349-9b63-424d-974e-327540e23ca6",
            AppKey = "0a166070-da06-4caf-9f51-7f72cae5e5bd"
        },
        new WeatherStationDataConnection{
            StreamId = "WeatherData-Berkeley",
            StreamName = "Berkeley Weather Station",
            ApiKey = "ca970349-9b63-424d-974e-327540e23ca6",
            AppKey  = "0a166070-da06-4caf-9f51-7f72cae5e5bd"
        },
        new WeatherStationDataConnection{
            StreamId = "WeatherData-Oakland",
            StreamName = "Oakland Weather Station",
            ApiKey = "ca970349-9b63-424d-974e-327540e23ca6"
```

We need more data! - Options

- Store device location info in the stream metadata

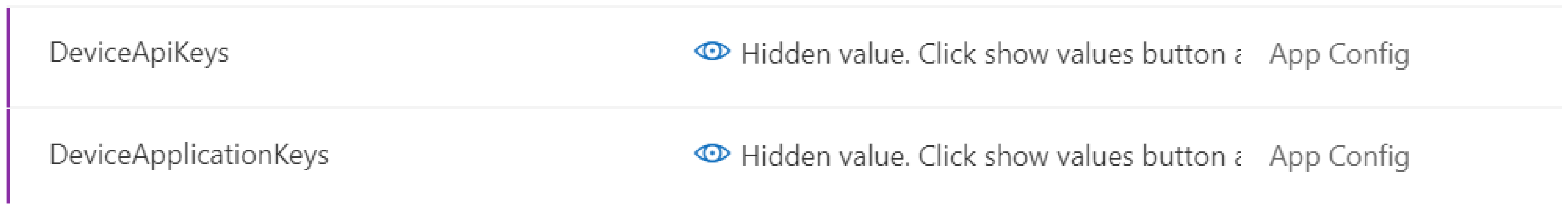


Stream Metadata

Device API Key	ca970349-9b63-424d-974e-327540e23ca6
Device Application Key	0a166070-da06-4caf-9f51-7f72cae5e5bd

We need more data! - Options

- Azure Function Configuration



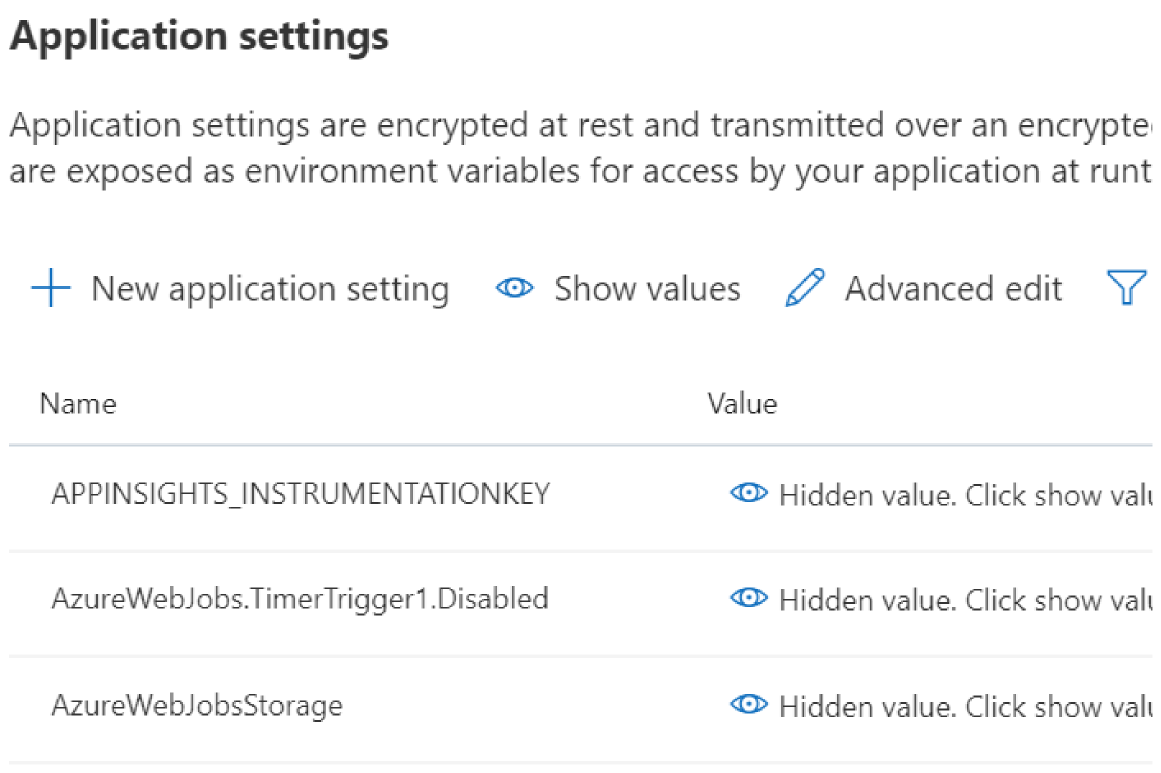
DeviceApiKeys	Hidden value. Click show values button a	App Config
DeviceApplicationKeys	Hidden value. Click show values button a	App Config

What Can We Do with the Data in OCS?

- OCS portal, Stream Explorer
- Stream metadata, manual or automatic
- View data via Data Views
- Trending 📈

Enhancements for Next Time

- Soft-code the secrets



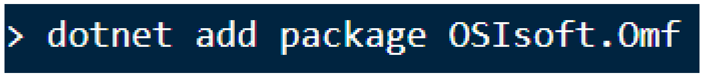
Application settings

Application settings are encrypted at rest and transmitted over an encrypte
are exposed as environment variables for access by your application at runt

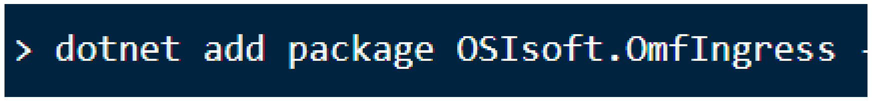
Name	Value
APPINSIGHTS_INSTRUMENTATIONKEY	Hidden value. Click show val
AzureWebJobs.TimerTrigger1.Disabled	Hidden value. Click show val
AzureWebJobsStorage	Hidden value. Click show val

Enhancements for Next Time

- Send data to OCS **via OMF**



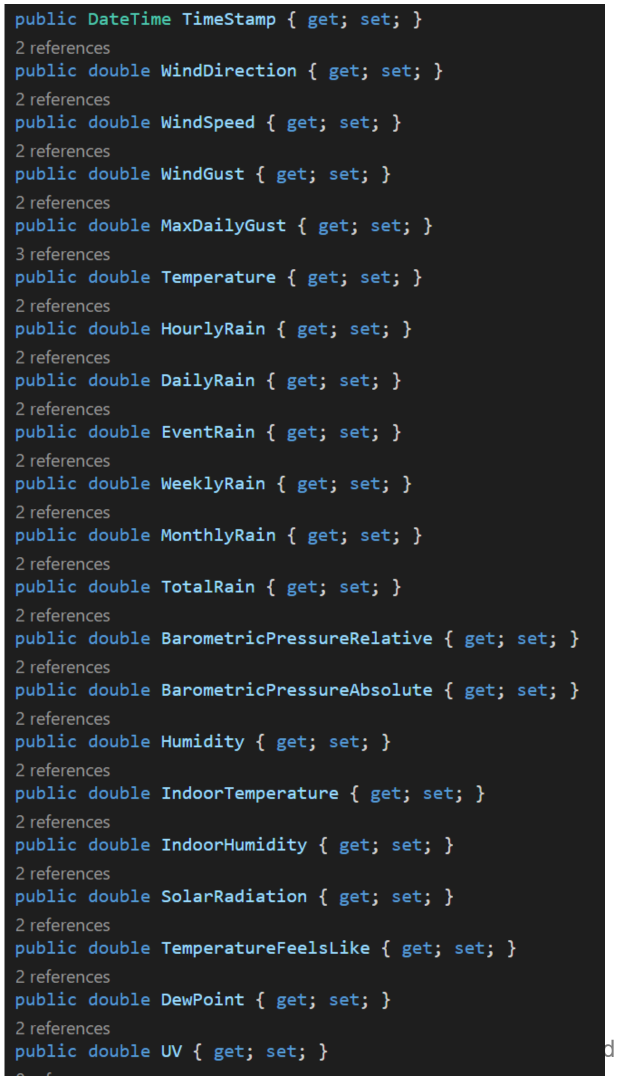
```
> dotnet add package OSIsoft.Omf
```



```
> dotnet add package OSIsoft.OmfIngress -
```

Enhancements for Next Time

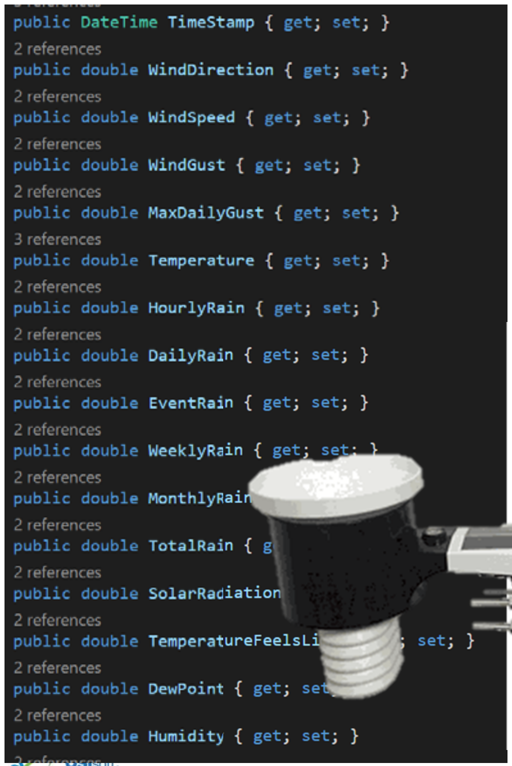
- Design the OCS Types better



```
public DateTime TimeStamp { get; set; }
2 references
public double WindDirection { get; set; }
2 references
public double WindSpeed { get; set; }
2 references
public double WindGust { get; set; }
2 references
public double MaxDailyGust { get; set; }
3 references
public double Temperature { get; set; }
2 references
public double HourlyRain { get; set; }
2 references
public double DailyRain { get; set; }
2 references
public double EventRain { get; set; }
2 references
public double WeeklyRain { get; set; }
2 references
public double MonthlyRain { get; set; }
2 references
public double TotalRain { get; set; }
2 references
public double BarometricPressureRelative { get; set; }
2 references
public double BarometricPressureAbsolute { get; set; }
2 references
public double Humidity { get; set; }
2 references
public double IndoorTemperature { get; set; }
2 references
public double IndoorHumidity { get; set; }
2 references
public double SolarRadiation { get; set; }
2 references
public double TemperatureFeelsLike { get; set; }
2 references
public double DewPoint { get; set; }
2 references
public double UV { get; set; }
```

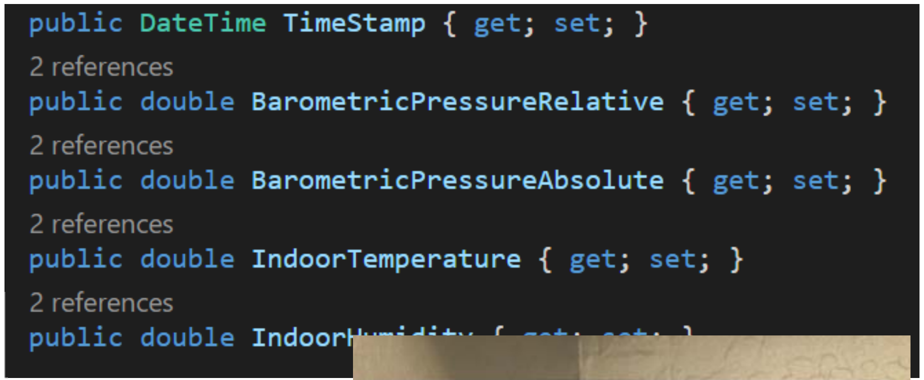
Weather station - streams

Weather Outdoor



```
public DateTime TimeStamp { get; set; }
2 references
public double WindDirection { get; set; }
2 references
public double WindSpeed { get; set; }
2 references
public double WindGust { get; set; }
2 references
public double MaxDailyGust { get; set; }
3 references
public double Temperature { get; set; }
2 references
public double HourlyRain { get; set; }
2 references
public double DailyRain { get; set; }
2 references
public double EventRain { get; set; }
2 references
public double WeeklyRain { get; set; }
2 references
public double MonthlyRain
2 references
public double TotalRain { g
2 references
public double SolarRadiation
2 references
public double TemperatureFeelsLi          ; set; }
2 references
public double DewPoint { get; set
2 references
public double Humidity { get; set; }
```

Weather Indoor



```
public DateTime TimeStamp { get; set; }
2 references
public double BarometricPressureRelative { get; set; }
2 references
public double BarometricPressureAbsolute { get; set; }
2 references
public double IndoorTemperature { get; set; }
2 references
public double IndoorH
```

Other options in weather stations

La Crosse

Davis

Summary

- We bought a weather station
- Read data from vendor cloud service
- Write it to OCS
- Extend to include other devices
- Do data stuff and trending stuff with it

Questions?

Please wait for the **microphone**

State your **name & company**

Save the Date...

AMSTERDAM
October 26-29, 2020

OSIsoft. PI World

謝謝
KEA LEBOHA
TAPADH LEIBH
고맙습니다
DZIĘKUJĘ CI
БАЯРЛАЛАА
MISAOTRA ANAO
NGIYABONGA
GRÀCIES
OBRIGADO
شكرا
SALAMAT
TEŞEKKÜR EDERIM
DANKIE
TERIMA KASIH
DANKON
TANK
TAPADH LEAT
KÖSZÖNÖM
СПАСИБО
MULȚUMESC
OSIsoft.
PI World
РАКМЕТ СІЗГЕ
FAAFETAI
HVALA
GO RAIBH MAITH AGAT
ESKERRIK ASKO
БЛАГОДАРЯ
GRACIAS
MAHADSANID
THANK YOU
HVALA
ХВАЛА ВАМ
ТИ БЛАГОДАРАМ
TEŞEKKÜR EDERIM
DANK JE
ΕΥΧΑΡΙΣΤΩ
GRATIAS TIBI
ДЗЯКУЙ
GRAZIE
TAK
DANKE
AČIŪ
SALAMAT
MAHALO IĀ ʻOE
TAKK SKAL DU HA
DI OU MÈSI
RAHMAT
MERCI
GRAZZI
ÞAKKA ÞÉR
FALEMINDERIT
ありがとうございました
ĎAKUJEM
HATUR NUHUN
РАХМАТ САҒА
SIPAS JI WERE
TERIMA KASIH
MATUR NUWUN
CẢM ƠN BẠN
UA TSAUG RAU KOJ
WAZVIITA
ТИ БЛАГОДАРАМ
СИПОС