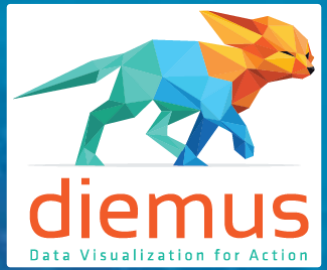
Just Another Weather Application Evaluating the OSIsoft Cloud Services

Lonnie A. Bowling



Old Programmers, where do they go?

They make Soylent out of us.

Recycled into yummy treats called "cheetos" and fed to proto-programmers. It's the circle of life.

You ever hear of Mountain Dew? It's old programmers.

OSIsoft Cloud Service - OCS
(formerly known as Qi)

- Early Days => QI 2015-2016
- Beta => 2017-2018
- General Availability === Today (I hope)!

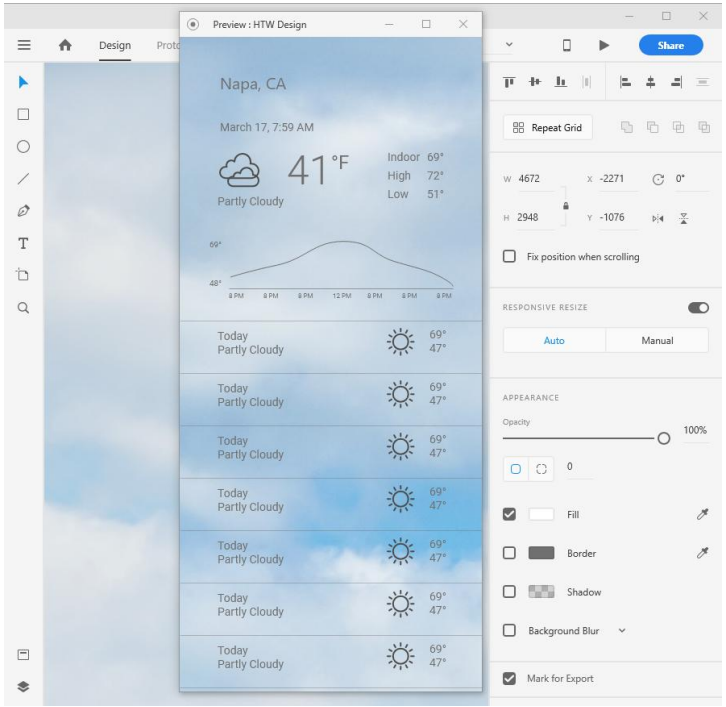
WARNING: NOT A DIRECT REPLACEMENT FOR YOUR PI SYSTEM

Evaluation Process

1. Find a real world use-case
2. Simple, but not too simple
3. Try to build and see what happens

How’s The Weather (HTW) is Born

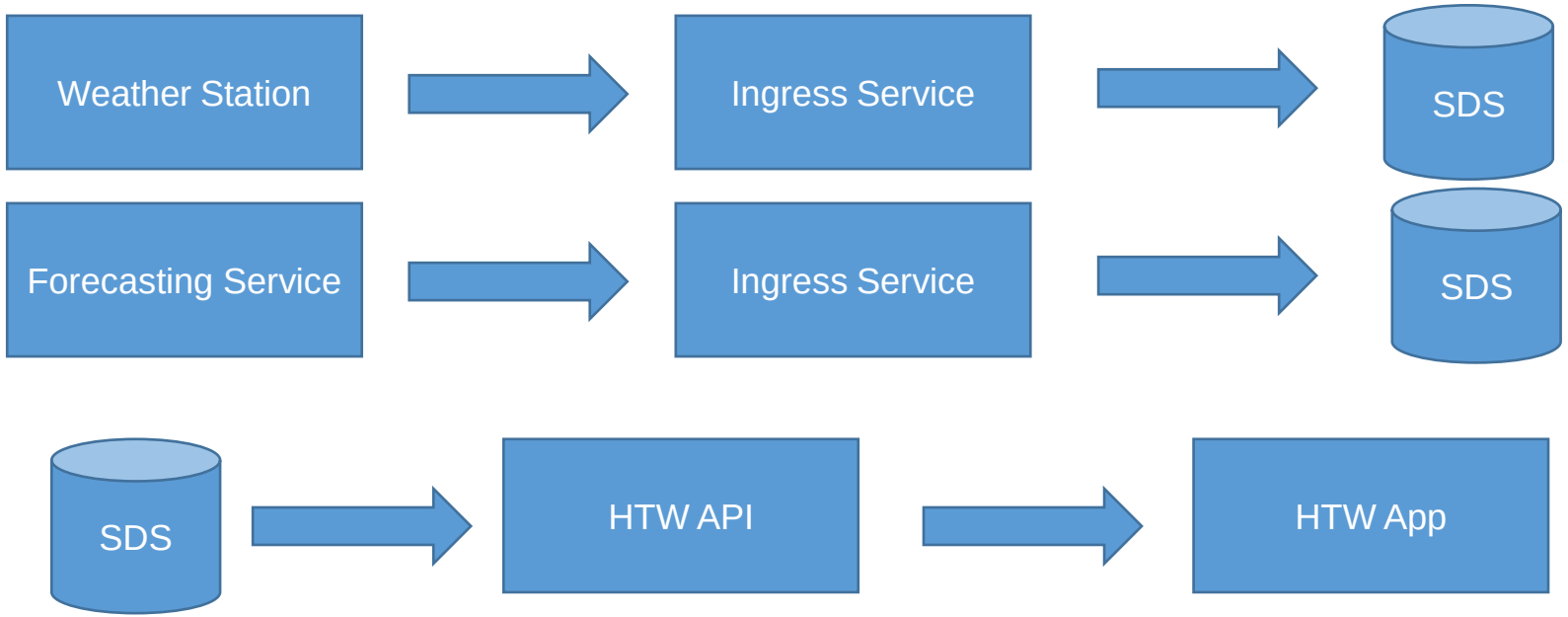
- Use a backyard weather station
- Use a weather forecasting service
- Mobile first
- Something that I would actually find useful



OCS – Critical Concepts

- SDS – Sequential Data Store
- Types – Strongly Typed database
- Stream – Each stream has a type

Architecture



The Weather Station

Ambient Weather WS-2902

Bad UI Design

The Weather Station – Data Request

https://api.ambientweather.net/v1/devices/macAddress?apiKey=&applicationKey=&endDate=&limit=288

…
"dateutc": 1515436500000,
"date": "2018-01-08T18:35:00.000Z",
"winddir": 58,
"windspeedmph": 0.9,
"tempf": 66.9,
"humidity": 30,
"baromrelin": 30.05,
"baromabsin": 28.71,
"tempinf": 74.1,
"humidityin": 30,
"hourlyrainin": 0,
"feelsLike": 66.9,
"dewPoint": 34.45380707462477
…

Forecasting Service – Dark Sky API

https://api.darksky.net/forecast/[key]/[latitude],[longitude]


```
 …
"daily": {
            "summary": "Mixed precipitation throughout the week.",
            "icon": "rain",
            "data": [{
                "time": 1509944400,
                "summary": "Rain starting in the afternoon, continuing until evening.",
                "icon": "rain",
                "sunriseTime": 1509967519,
                "sunsetTime": 1510003982,
                "precipIntensity": 0.0088,
                "precipIntensityMax": 0.0725,
                "precipIntensityMaxTime": 1510002000,
…
```

Data Ingress – The Plan

1. Azure Functions using .Net Core
2. Import C# OCS library
3. Configuring OCS
4. Reading Data from sources
5. Write Data to OCS
6. Note - Could have used OMF

Stream Types in OCS

- Streams are like PI Points
- Types are user defined
- Immutable (can’t be changed)
- Complex as you want
- Even nested

Defining Types – Complex

```
public class WeatherEvent
{
    [SdsMember(IsKey = true)]
    public DateTime Timestamp { get; set; }

    public float Temperature { get; set; }

    public float Humidity { get; set; }

    public float WindSpeed { get; set; }

    public string Summary { get; set; }

}
```

Defining Types – Simple

```
public class StreamDouble
{
    [SdsMember(IsKey = true)]
    public DateTime Timestamp { get; set; }
    public double Value { get; set; }
}

public class StreamBool
{
    [SdsMember(IsKey = true)]
    public DateTime Timestamp { get; set; }
    public bool Value { get; set; }
}
```

Defining Types – Complex or Simple? A fork in the road…

When you come to a fork in the road, take it - Yogi Berra

And the winner is…

Simple Types

- Types are immutable
- To change a type
  - Delete all streams using that type (all data is lost)
  - Delete the type
  - Create new type
  - Create new streams

Configuring OCS

- Create a Namespace
- Create a client account for the service

(Secret values are only accessible during creation time!)

- Give account write permissions

```
public class connectionInfo
    {
        public static string accountId = "d0e32177-f027-43f0-8bb9-c2e09401e555";
        public static string namespaceId = "htw";
        public static string resource = "https://dat-b.osisoft.com";
        public static string clientId = "bbf85b98-bfab-437b-a48d-b89a38eb4c93";
    }
```

DEMO

Data Ingress

HTW API

- Connect to OCS (like we did for data ingress)
- Provide three API Calls

```
http://localhost:58588/api/Weather/GetSnapshot

http://localhost:58588/api/Weather/GetTrend?id=home.pluto.tempf

http://localhost:58588/api/Weather/GetForecast
```

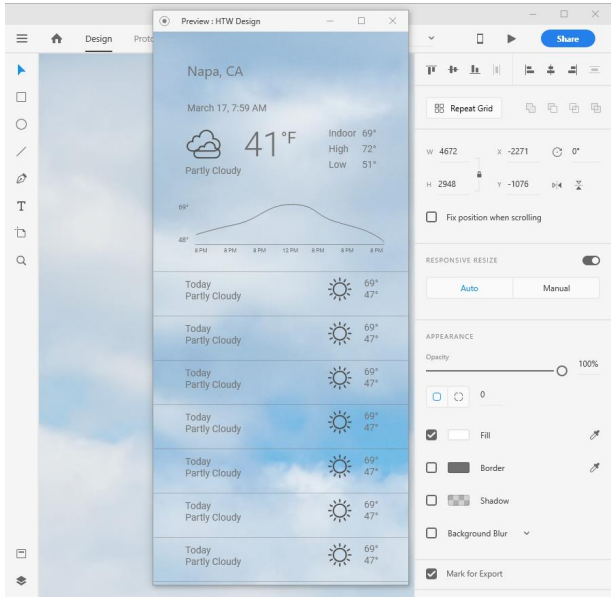
DEMO

HTW API

HTW Application

https://howstheweather.azurewebsites.net/

- Mobile First
- Current and forecast
- Simple, clean graphics
- Fast loading

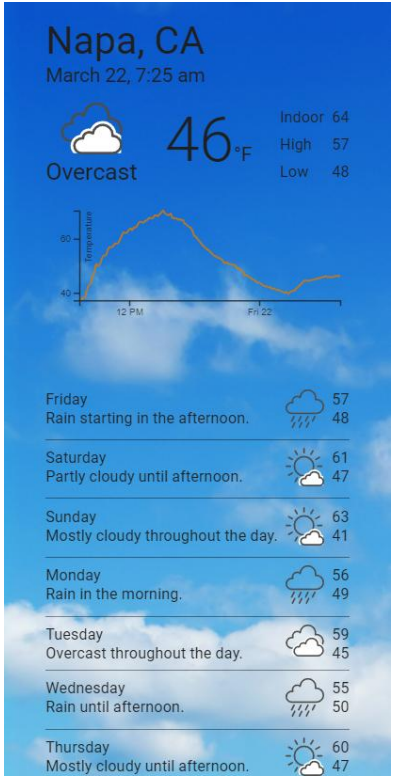


DEMO

HTW Application

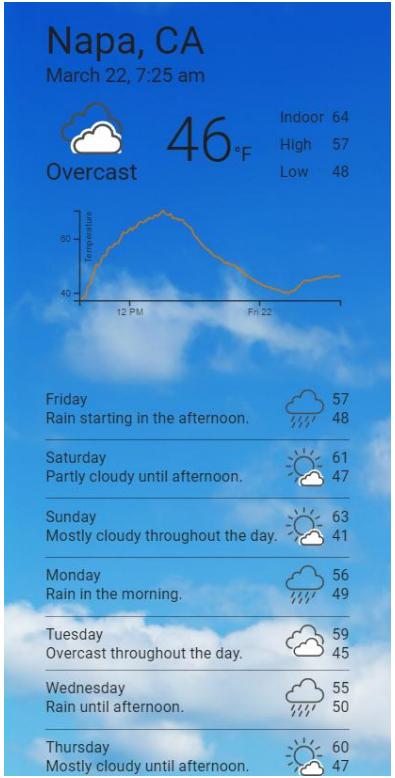
My overall experience

- Fairly Simple to get started
- Understanding types is critical
- All base methods you would expect are there
- Bulk queries capability, but limited to single stream types
- Unit of Measure limited, cannot add new UOMs



What is new

- OSIsoft's Identity Service
  - OAuth 2.0 & OpenID
  - Authentication by external identity provider
  - Authorization handled by configuring roles
- Data Views (Preview) – customizable tables
  - Streams properties mapped to table columns



What you will need to bring

- Data structure (no AF like feature)
- Analytics Engine

Thanks for Attending!

Lonnie A. Bowling

Data Interaction Expert

Diemus, Inc.

lonnie@diemus.com

diemus

Data Visualization for Action

Application Demo Site: https://howstheweather.azurewebsites.net/

Source Code: https://github.com/LonnieBowling/howstheweather

Questions?

Please wait for the **microphone**

State your **name & company**

Please remember