

MQTT Location Geodaten REST Entität Zeitreihe API OGC Echtzeit Datensatz SensorNetzwerk Datenmodell Datenstream Umweltmonitoring Sensor Things Beobachtung Datenerfassung Sensor IoT Thing SWE JSON FeatureOfInterest Geoserver OpenAPI

## 28. QGIS-Talk

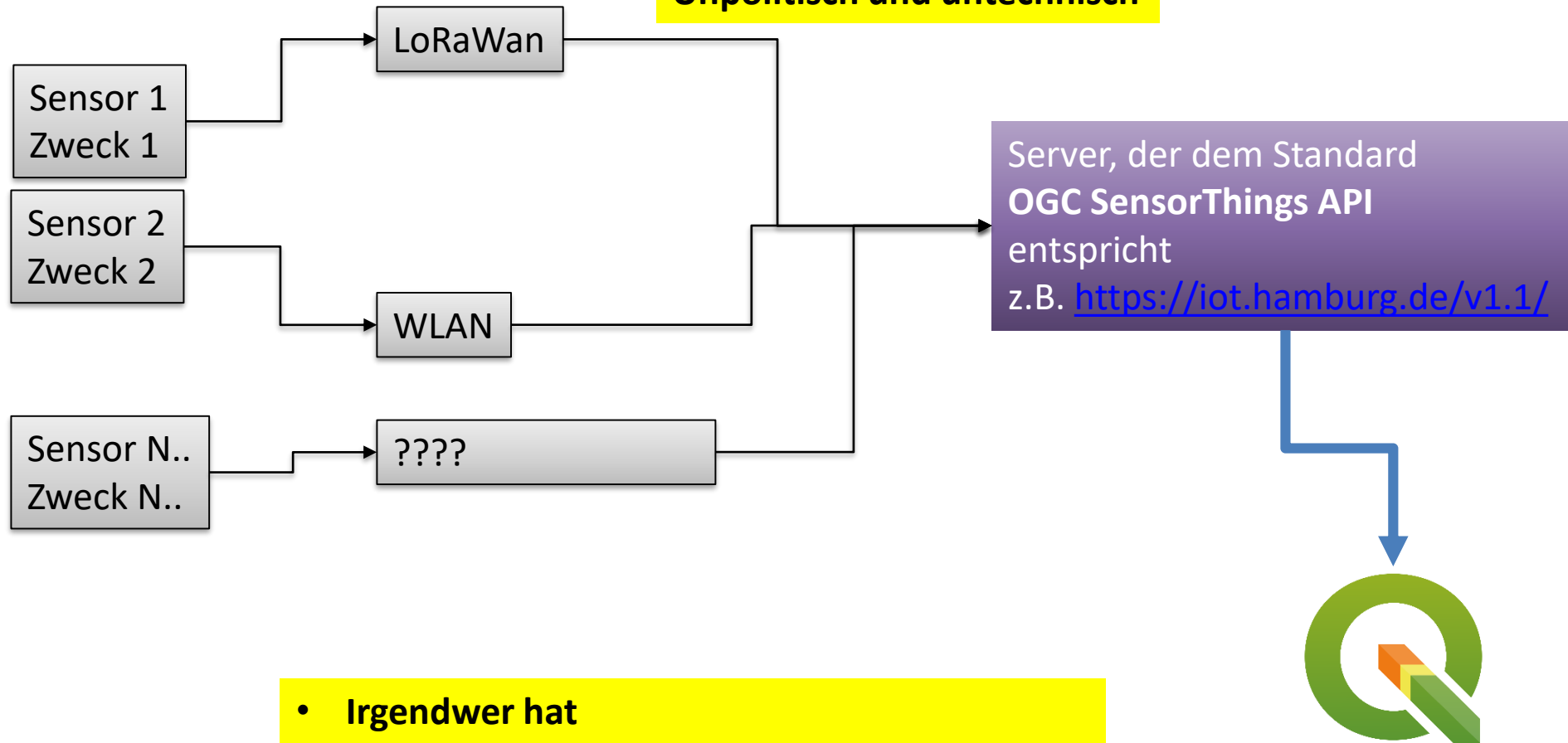
### Sensor Things API in QGIS

...erste Schritte

# Zuviel Input



## Unpolitisch und untechnisch



- Irgendwer hat
- eine unbekannte Anzahl an Sensoren
- mit unbekannten Zwecken,
- die auf einem OGC SensorThings API - Server
- gesammelt und bereitgestellt werden.....

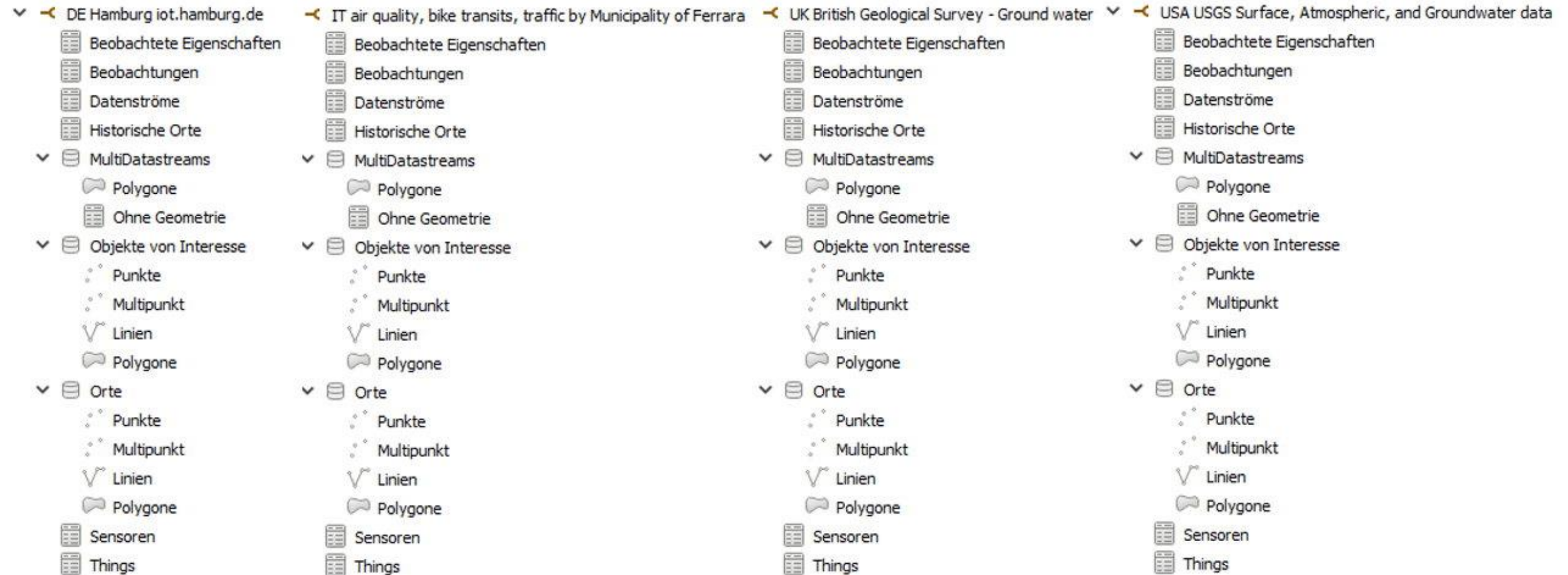




**FROST (FRaunhofer Open Source SensorThings) Server**

die bekannteste Umsetzung des Standards **OGC SensorThings API**

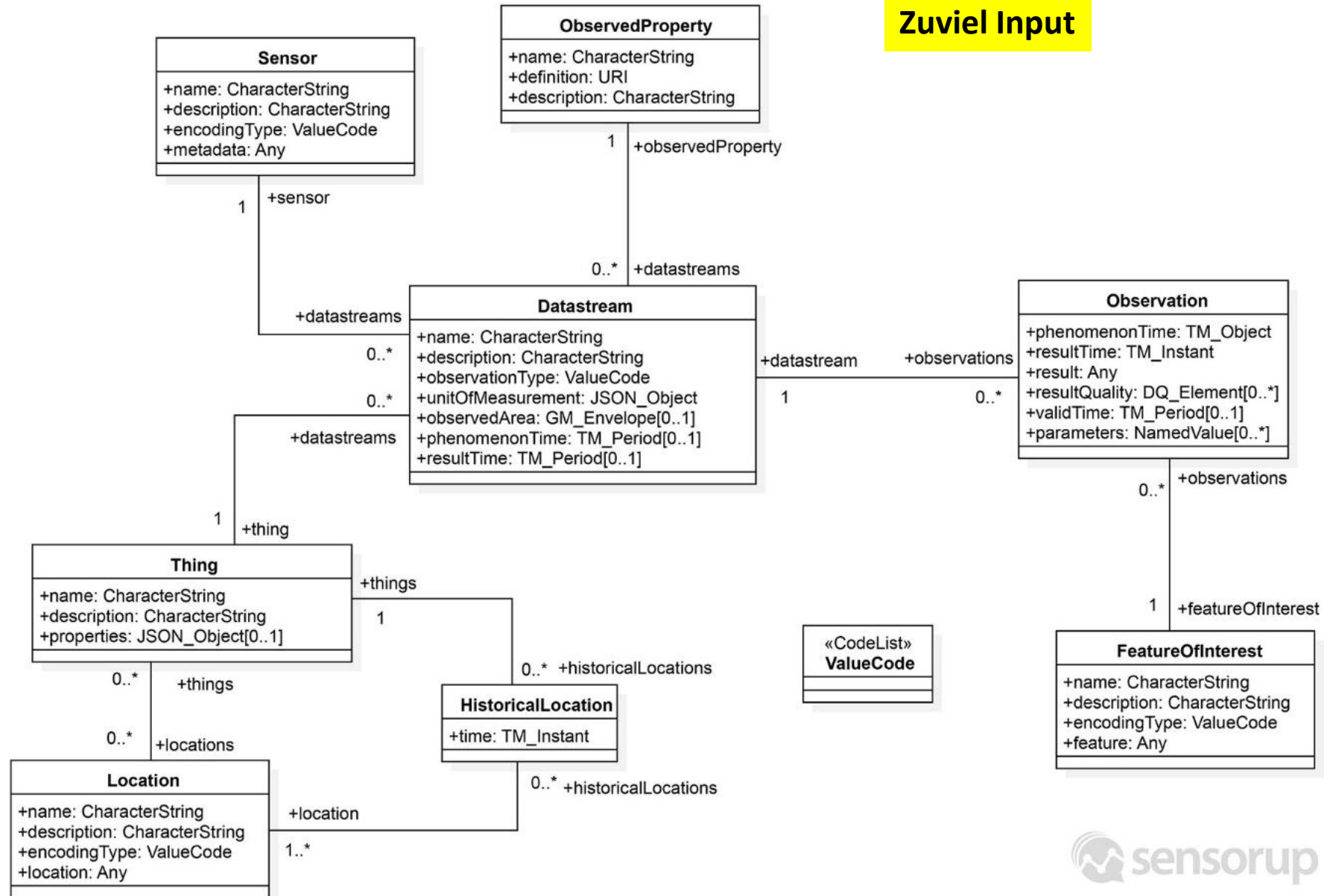
## Tipp 1: alle Server sind gleich aufgebaut und werden strukturell übersetzt



**Tipp 2: wieviel wo drin ist, kann sehr unterschiedlich und viel sein**

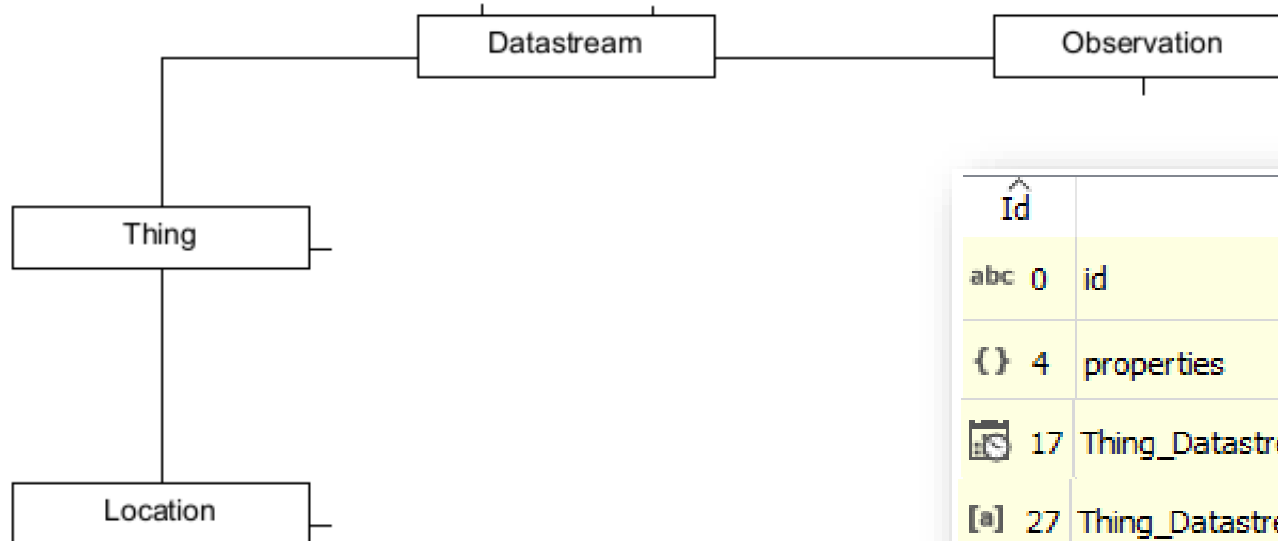
**Tipp 3: Wir machen GIS – also starten wir immer mit Locations/Orte**

Entity	Hamburg	Italien	UK	USA
Locations	5.463	917	131	18.647
HistoricalLocations	2.894	1	4.221	11.218.413
Things	2.687	1.113	102	18.646
Datastreams	7.207	2.945	278	40.065
Sensors	12	47	12	1
ObservedProperties	22	68	34	315
Observations	255.127.100	16.691.293	10.421.432	47.340.392
FeaturesOfInterest	11.172.500	917	233	19.471



**Tipp 4: die Verbindung Location-Thing-Datastream-Observation reicht (fast) immer**

**Tipp 5: es gibt 4 Datentypen ☹ ... aber mit Beispielen geht das auch 😊**



Id	Name	Alias	Typ	Typname
abc 0	id		Text (string)	
{ } 4	properties		Karte	json
🕒 17	Thing_Datastream_phenomenonTimeStart		Datum & Zeit	
[a] 27	Thing_Datastream_Observation_resultQuality		Zeichenkettenliste	



## Praktischer Teil 1: erstelle eine Karte aller Sensoren – thematisch aufgeteilt

1. <https://iot.hamburg.de/v1.1/> als SensorThings eintragen
2. Aufklappen und Orte – Punkte auswählen
3. Karte ist fertig ..... Was ist drin
4. Mit F6 die Attributtabelle öffnen
5. In der Regel steht (nur) im Feld name oder description was brauchbares drin
6. Durch scrollen/sortieren einen Überblick verschaffen  
(Profis: Werkzeug „Statistik nach Kategorie“)
7. Ein guter Start ist es, mit den ersten Zeichen zu starten, also kategorisierte Darstellung nach left(“name“,10)

Siehe auch [Wiki-Dokumentation](#) : die erste Karte "alle Sensoren"

## Praktischer Teil 2: erstelle eine Karte mit einer Sensor-Kategorie und allen Things

1. <https://iot.hamburg.de/v1.1/> über Layer - Datenquellenverwaltung - SensorThings - "Hamburg,, öffnen
2. Orte (als Entity-Typ) und Punkt (als Geometrietyp) auswählen und hinzufügen = alle Orte
3. Filter für die Einschränkung befüllen
  1. **substringof('kehrsz',name)** filtert alle Objekte, die die Zeichenkette **kehrsz** im Attribut **name** enthalten
  2. **startswith(name,'Verkehrs')** filtert alle Objekte, die mit der Zeichenkette **Verkehrs** im Attribut **name** beginnen
  3. **name eq 'Verkehrszählstelle 0343932'** filtert alle Objekte, die die Zeichenkette **Verkehrszählstelle 0343932** im Attribut **name** enthalten
4. hinzufügen = alle Orte, die den Filter erfüllen
5. erweitern der Auswahl auf Things
6. Ergebnis ist weiter ein Punktlayer, jedoch mit nur noch mit den Punkten, die ein Thing haben
7. Die Attribute sind jetzt zusammengesetzt aus den Orten und den Things (erkennbar am Prefix)

Siehe auch [Wiki-Dokumentation](#) : die zweite Karte "eine Sensor-Sorte mit Things"

### Praktischer Teil 3: erstelle eine Karte mit „einem Sensor mit seinen Datenströmen“

1. <https://iot.hamburg.de/v1.1/> über Layer - Datenquellenverwaltung - SensorThings - "Hamburg,, öffnen
2. Orte (als Entity-Typ) und Punkt (als Geometrietyp) auswählen und hinzufügen = alle Orte
3. Filter für die Einschränkung befüllen
  1. `substringof('kehrsz',name)` filtert alle Objekte, die die Zeichenkette **kehrsz** im Attribut **name** enthalten
  2. `startswith(name,'Verkehrs')` filtert alle Objekte, die mit der Zeichenkette **Verkehrs** im Attribut **name** beginnen
  3. **`name eq 'Verkehrszählstelle 0343932'`** filtert alle Objekte, die die Zeichenkette **Verkehrszählstelle 0343932** im Attribut **name** enthalten
4. erweitern der Auswahl auf Thing
5. erweitern der Auswahl auf Datenstrom
6. Ergebnis ist weiter ein Punktlayer, jedoch mit nur noch genau einem Punkt mit (allen) Thing und allen Datenströmen
7. Sichten der Datenströme durch kategorisierte Darstellung nach „Thing\_Datastream\_description“

Siehe auch [Wiki-Dokumentation](#) : die dritte Karte "ein Sensor mit seinen Datenströmen"

#### **Praktischer Teil 4: "ein Sensor mit einem bestimmten Datenstrom und allen Werten aus einem Jahr"**

1. <https://iot.hamburg.de/v1.1/> über Layer - Datenquellenverwaltung - SensorThings - "Hamburg,, öffnen
2. Orte (als Entity-Typ) und Punkt (als Geometrietyp) auswählen und hinzufügen = alle Orte
3. Filter für die Einschränkung befüllen
  1. `substringof('kehrsz',name)` filtert alle Objekte, die die Zeichenkette **kehrsz** im Attribut **name** enthalten
  2. `startswith(name,'Verkehrs')` filtert alle Objekte, die mit der Zeichenkette **Verkehrs** im Attribut **name** beginnen
  3. **`name eq 'Verkehrszählstelle 0343932'`** filtert alle Objekte, die die Zeichenkette **Verkehrszählstelle 0343932** im Attribut **name** enthalten
4. erweitern der Auswahl auf Thing
5. erweitern der Auswahl auf Datenstrom
6. Filtern auf **`substringof('Tag',name)`**
7. erweitern der Auswahl auf Beobachtungen (über Begrenzung nachdenken!)
8. Filtern auf **`year(resultTime) eq 2025`**
9. Ergebnis ist weiter ein Punktlayer, jedoch mit nur noch genau einem Punkt mit (allen) Thing und einem Datenstrom und allen Beobachtungen aus 2025

Siehe auch [Wiki-Dokumentation](#) : die vierte Karte "ein Sensor mit einem bestimmten Datenstrom und allen Werten aus einem Jahr"

## Praktischer Teil 5: " das Diagramm zur vierten Karte"

1. <https://iot.hamburg.de/v1.1/> über Layer - Datenquellenverwaltung - SensorThings - "Hamburg,, öffnen
2. Orte (als Entity-Typ) und Punkt (als Geometrietyp) auswählen und hinzufügen = alle Orte
3. Filter für die Einschränkung befüllen
  1. `substringof('kehrsz',name)` filtert alle Objekte, die die Zeichenkette **kehrsz** im Attribut **name** enthalten
  2. `startswith(name,'Verkehrs')` filtert alle Objekte, die mit der Zeichenkette **Verkehrs** im Attribut **name** beginnen
  3. **`name eq 'Verkehrszählstelle 0343932'`** filtert alle Objekte, die die Zeichenkette **Verkehrszählstelle 0343932** im Attribut **name** enthalten
4. erweitern der Auswahl auf Thing
5. erweitern der Auswahl auf Datenstrom
6. Filtern auf **`substringof('Tag',name)`**
7. erweitern der Auswahl auf Beobachtungen (über Begrenzung nachdenken!)
8. Filtern auf **`year(resultTime) eq 2025`**
9. Ergebnis ist weiter ein Punktlayer, jedoch mit nur noch genau einem Punkt mit (allen) Thing und einem Datenstrom und allen Beobachtungen aus 2025
10. über Erweiterungen - Erweiterungen verwalten und installieren - Data Plotly installieren und Diagramm erstellen über result und resulttime

Siehe auch [Wiki-Dokumentation](#) : das Diagramm zur vierten Karte



## Praktischer Teil 6: " zeige alle Sensoren einer Art mit dem aktuellen Sensor-Wert"

1. <https://iot.hamburg.de/v1.1/> über Layer - Datenquellenverwaltung - SensorThings - "Hamburg,, öffnen
2. Orte (als Entity-Typ) und Punkt (als Geometrietyp) auswählen und hinzufügen = alle Orte
3. Filter für die Einschränkung befüllen **substringof('kehrsz',name) and substringof('uerschnitt',description)**
4. erweitern der Auswahl auf Thing
5. erweitern der Auswahl auf Datenstrom
6. Filtern auf **substringof(,15',name)**
7. erweitern der Auswahl auf Beobachtungen
8. Filtern auf **Begrenzung=1** und Sortieren nach **resultTime** und Sortierung **absteigend**
9. Ergebnis ist ein Punktlayer mit dem aktuellen/letzten Wert des Datenstroms

Siehe auch [Wiki-Dokumentation](#) : zeige alle Sensoren einer Art mit dem aktuellen Sensor-Wert