GraphQL – A New Method for Instant Data Insights on Haystack

Patrick Coffey – WideSky
Agenda

1. Haystack API today
2. GraphQL
3. GraphQL on Haystack Examples
4. Demos
Haystack API Today
API clients

Haystack Server

Tagging

Time-series Data

Application users

Integrated Systems

May 13-15, 2019
What was yesterday’s max power readings for each space at the head office?

http://haystack/
1: Find Spaces

GET

http://haystack/read

Response

We need these ID’s

```
ver:"3.0"
filter,limit
"space and siteRef==@vrt",30

ver:"3.0"
id,dis,primaryFunction,siteRef,space
@vrt.dc "dc","Data Centre","Data centre",@vrt "vrt",M
@vrt.gf "gf","Ground Floor","commercial",@vrt "vrt",M
@vrt.l1 "l1","Level 1","commercial",@vrt "vrt",M
```
2: Find points

For each ID
@vrt.dc
@vrt.gf
@vrt.l1

GET

http://haystack/read

Response

ver:"3.0"
id,dis,elec,equipRef,his,kind,point,power,sensor,siteRef,tz,unit
@vrt.dc_ac1.elecPower"elecPower","Active Power Total",M,@vrt.dc_ac1"dc_ac1",M,"Number",M,M,M,@vrt "vrt","GMT-10","kW"
@vrt.dc_ups.elecPower"elecPower","Active Power Total",M,@vrt.dc_ups"dc_ups",M,"Number",M,M,M,@vrt "vrt","GMT-10","kW"

May 13-15, 2019
For each ID

**vr.t.dc**
- @vr.t.dc_ac1.elecPower
- @vr.t.dc_ups.elecPower

**vr.t.gf**
- @vr.t.gf.ac1
- @vr.t.gf.ac2
- @vr.t.gf.gl1
- @vr.t.gfgp1
- @vr.t.gf.gp2
- @vr.t.gf.hw1

**vr.t.l1**
- @vr.t.gf.ac1
- @vr.t.gf.ac2
- @vr.t.gf.gl1
- @vr.t.gf.gl2
- @vr.t.gf.gp1

...
4: Display results

-vrt.dc
  @vrt.dc_ac1.elecPower 1.1
  @vrt.dc_ups.elecPower 1.2

-vrt.gf
  @vrt.gf.ac1 1.3
  @vrt.gf.ac2 0.1
  @vrt.gf.gl 0.5
  @vrt.gf.gp1 1.5
  @vrt.gf.gp2 2.2
  @vrt.gf.hw1 4.6

-vrt.l1
  @vrt.l1.ac1 7.9
  @vrt.l1.ac2 0.6
  @vrt.l1.gl1 0.8
  @vrt.l1.gl2 1.2
  @vrt.l1.gp1 1.5

...
Haystack REST API

- Simple data exchange
  - Full data fidelity
  - All timeseries data
- Plenty of open source tools
- Real-time monitoring and control

- Back-and-forth communications
- Data Over-Fetching
- Hard for non-devs to build insights
- Underutilized Haystack servers
Our API Needs

• Build insights from the browser
• Simple to learn
• Nested query
• Leverage existing server capabilities
Why didn’t you just use...

<table>
<thead>
<tr>
<th>SPARQL</th>
<th>Cypher</th>
<th>SQL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extend REST ops</td>
<td>Client side scripts</td>
<td>Server side scripts</td>
</tr>
<tr>
<td>RDF</td>
<td>OData</td>
<td>OWL</td>
</tr>
</tbody>
</table>
GraphQL
• Specification open-sourced by Facebook 2015
• A query language for APIs
• Span many resources
• Type system - server describes what's possible
• Significant open source community
GraphQL

```graphql
query {
  allPeople(first: 1) {
    people {
      name
      gender
    }
  }
}
```

POST

http://starwars/graphql

Response

```
{
  "data": {
    "allPeople": {
      "people": [
        {
          "name": "Luke Skywalker",
          "gender": "male"
        }
      ]
    }
  }
}
```
Our Approach

- Simple static schema
- Leverage nesting capability
- Expose timeseries functions to user
- Complement Haystack REST API
What are the tags on entity x?

```json
{
  "data": {
    "haystack": {
      "entity": {
        "tags": [
          {
            "name": "dis",
            "value": "Active Energy Import"
          },
          {
            "name": "elec",
            "value": "marker"
          },
          {
            "name": "energy",
            "value": "marker"
          },
          {
            "name": "equipRef",
            "value": "782d7bd4-6bcc-11e9-838a-0242ac120004"
          },
          {
            "name": "fqname",
            "value": "vrt.gf_inc.elecEnergy"
          },
          {
            "name": "his",
            "value": "marker"
          },
          {
            "name": "hisTotalized",
            "value": "marker"
          },
          {
            "name": "id",
            "value": "b97b1bf0-6bcc-11e9-85a6-0242ac120004"
          }
        ]
      }
    }
  }
}
```
What was yesterday's history for entity x?

```json
{
    "data": {
        "haystack": {
            "entity": {
                "history": {
                    "timeSeries": [
                        {
                            "dataPoints": [
                                {
                                    "time": 15572889000000,
                                    "value": "241863.798"
                                },
                                {
                                    "time": 15572890000000,
                                    "value": "241866.842"
                                },
                                {
                                    "time": 15572907000000,
                                    "value": "241869.807"
                                },
                                {
                                    "time": 15572916000000,
                                    "value": "241872.884"
                                },
                                {
                                    "time": 15572925000000,
                                    "value": "241875.861"
                                }
                            ]
                        }
                    ]
                }
            }
        }
    }
}
```
Find all spaces at the head office

```json
{
    "data": {
        "haystack": {
            "search": {
                "count": 3,
                "entity": [
                    {
                        "id": "6dee3c9e-6bcc-11e9-8329-0242ac120004",
                        "description": "Data Centre"
                    },
                    {
                        "id": "6e95524e-6bcc-11e9-8329-0242ac120004",
                        "description": "Ground Floor"
                    },
                    {
                        "id": "6e2440d2-6bcc-11e9-832b-0242ac120004",
                        "description": "Level 1"
                    }
                ]
            }
        }
    }
}
```
Nesting

# requests vs. layers

- Black line: Haystack REST API
- Blue line: GraphQL

May 13-15, 2019
For each space, get yesterday's power meter data.

```json
{
  "data": {
    "haystack": {
      "search": {
        "entity": [
          {
            "description": "Data Centre",
            "search": {
              "history": {
                "timeSeries": [
                  {
                    "dataPoints": [
                      {
                        "time": 1557289260000,
                        "value": "0.09"
                      },
                      {
                        "time": 1557289320000,
                        "value": "0.09"
                      },
                      {
                        "time": 1557289380000,
                        "value": "0.09"
                      },
                      {
                        "time": 1557289440000,
                        "value": "0.09"
                      }
                    ]
                  }
                ]
              }
            }
          }
        ]
      }
    }
  }
}
```
Add description from equipRef and compute max() for each
1: Developer IDE - GraphiQL
```graphql
query {

}
```
2: Data interaction in Grafana
Summary

• Benefits:
  • Users become data superheroes from the browser
  • Haystack adoption
  • Other use cases – RBAC rules
• Extensions:
  • Auto unit conversion
  • Timeseries functions – (select, aggregate, transform...)
  • Entity filtering based on history
• Haystack v4?
Thank You

Come and say G’Day!