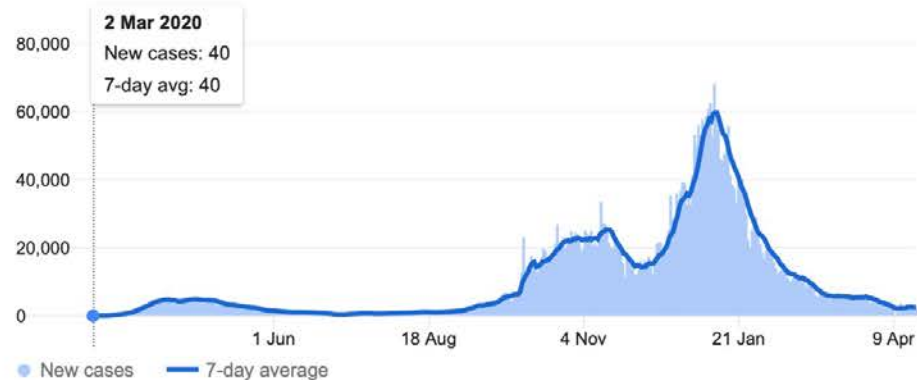


Downloading data using APIs

APIs provide access to data

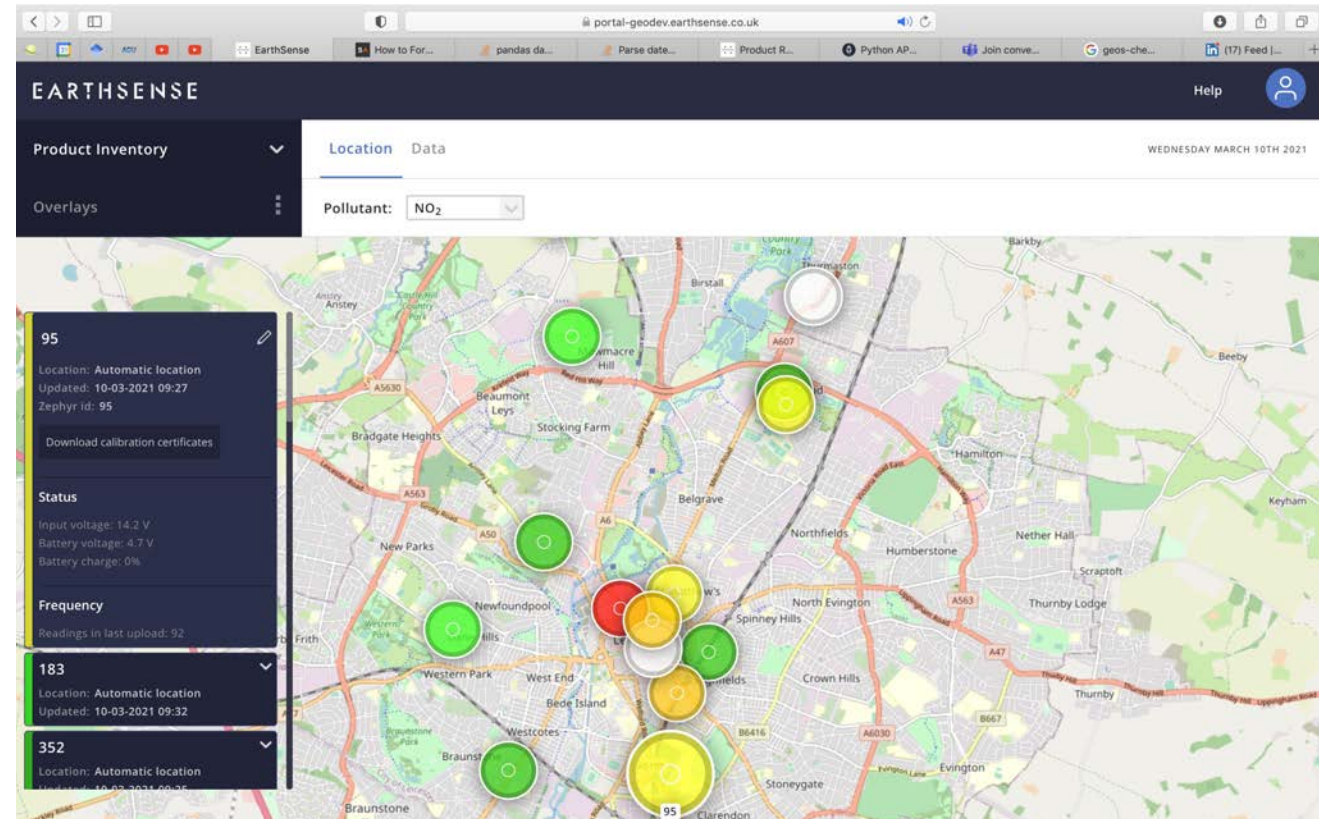
Greater control of what portion of the data
you are retrieving

To use live information (a
database that is constantly be
updated)



Each day shows new cases reported since the previous day · [About this data](#)

The data is too large to download from the website



Resources

<https://www.dataquest.io/blog/python-api-tutorial/>

- Required Python packages and how to install them
- example APIs
- How to manipulate the resulting data (JSON, yuck!)

API Server

```
graph TD; PS[Python script] -- "'GET' Request" --> AS[API Server]; AS --> RC[Response Code]; AS --> RD[Response Data];
```

The diagram illustrates an API interaction. At the top is a grey rectangular box labeled 'API Server'. Below it, on the left, is a blue rectangular box labeled 'Python script'. A blue arrow points upwards from the 'Python script' box to the 'API Server' box, with the text ''GET' Request' written in blue next to the arrow. From the bottom of the 'API Server' box, a green line descends and then splits into two separate green arrows pointing downwards. The left arrow is labeled 'Response Code' and the right arrow is labeled 'Response Data' in green text.

'GET'
Request

Python script

Response
Code

Response
Data

API Status Codes (response codes)

- 200: Everything went okay, and the result has been returned (if any).
- 301: The server is redirecting you to a different endpoint. This can happen when a company switches domain names, or an endpoint name is changed.
- 400: The server thinks you made a bad request. This can happen when you don't send along the right data, among other things.
- 401: The server thinks you're not authenticated. Many APIs require login credentials, so this happens when you don't send the right credentials to access an API.
- 403: The resource you're trying to access is forbidden: you don't have the right permissions to see it.
- 404: The resource you tried to access wasn't found.
- 503: The server is not ready to handle the request.

Be careful, they work for the example APIs used in the tutorial here, but could not be relied on when I used an API from an air pollution database

JavaScript Object Notation (JSON) data (response data)

- JSON is the language of the API (a requests for data returns the data in this format/language)
- JSON is a way to encode data structures that ensures that they are easily readable by machines.
 - But difficult to work with (for me!)
- Use the Python package, 'json', to convert the JSON to a more familiar object

```
def jprint(obj):  
    # create a formatted string of the Python JSON object  
    text = json.dumps(obj, sort_keys=True, indent=4)  
    print(text)
```

```
[  
  {  
    "name": "Sabine",  
    "age": 36,  
    "favorite_foods": ["Pumpkin", "Oatmeal"]  
  },  
  {  
    "name": "Zoe",  
    "age": 40,  
    "favorite_foods": ["Chicken", "Pizza", "Chocolate"]  
  },  
  {  
    "name": "Heidi",  
    "age": 40,  
    "favorite_foods": ["Caesar Salad"]  
  }  
]
```

Working with JSON was difficult for me, and the structure of the data was not consistent – across different APIs, as well as within the same API (but for different time periods)

Example API in the tutorial...

API Server

'GET'
Request

Python script

Retrieve
data

endpoint

JavaScript
Object
Notation

```
response = requests.get("http://api.opennotify.org/astros.json")
```

Response data (JSON format) *jprint(response)*

```
'message': 'success', 'number': 7,  
'people': [{ 'craft': 'ISS', 'name':  
'Sergey Ryzhikov'}, { 'craft': 'ISS',  
'name': 'Kate Rubins'}, { 'craft': 'ISS',  
'name': 'Sergey Kud-Sverchkov'},  
{ 'craft': 'ISS', 'name': 'Mike  
Hopkins'}, { 'craft': 'ISS', 'name':  
'Victor Glover'}, { 'craft': 'ISS',  
'name': 'Shannon Walker'}, { 'craft':  
'ISS', 'name': 'Soichi Noguchi' } ] }
```

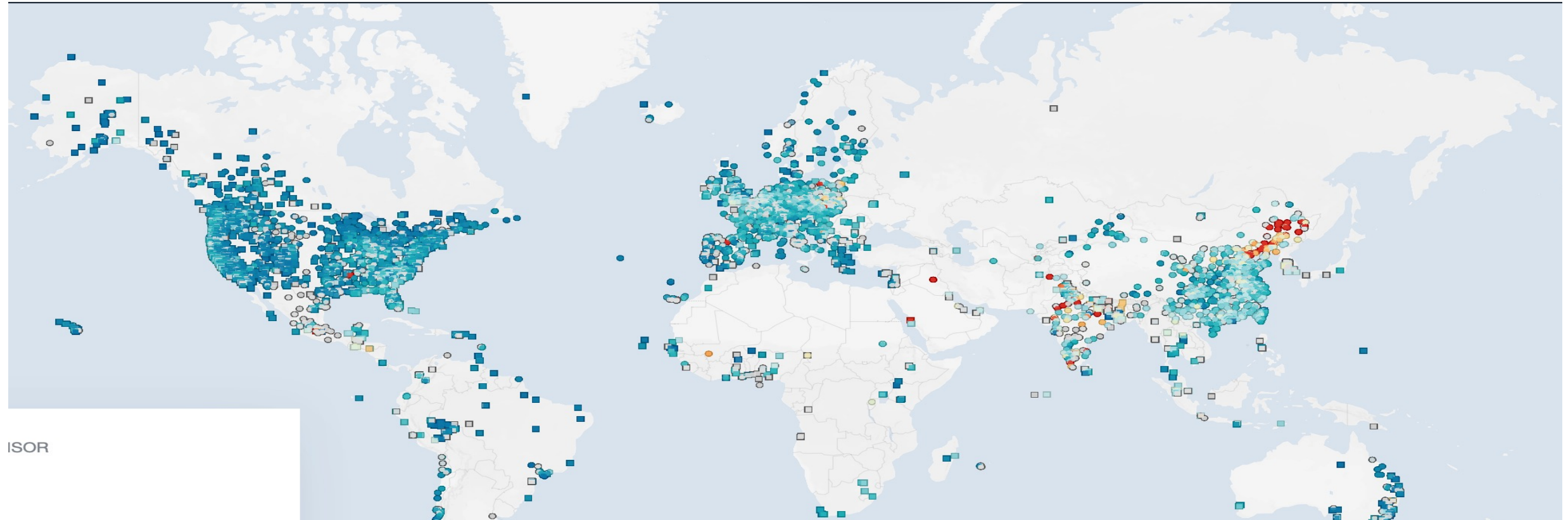
Response data (in the form of a string)

```
{  
  "message": "success",  
  "number": 6,  
  "people": [  
    {  
      "craft": "ISS",  
      "name": "Alexey Ovchinin"  
    },  
    {  
      "craft": "ISS",  
      "name": "Nick Hague"  
    },  
    {  
      "craft": "ISS",  
      "name": "Christina Koch"  
    },  
    {  
      "craft": "ISS",  
      "name": "Alexander Skvortsov"  
    },  
    {  
      "craft": "ISS",  
      "name": "Luca Parmitano"  
    },  
    {  
      "craft": "ISS",  
      "name": "Andrew Morgan"  
    }  
  ]  
}
```


Next example API in the tutorial introduces an API that uses a query parameter

- With these, you can have more control of what data you are requesting (e.g. time, location, pollutants, etc.)

Open AQ – A publicly available database of sensors around the world



- Info on Open AQ website on their own APIs is NOT USEFUL
- I found a nice Python wrapper, developed by David Hagan (w/ tutorial), which was much more useful
- One huge advantage to this package is that it loads in the form of Panda DataFrame, which is much more easier to handle than JSON
- <http://dhhagan.github.io/py-openaq/tutorial/api.html>
- <https://py-openaq.readthedocs.io/en/latest/>

```
data = api.latest(city='Delhi',
parameter='pm25', df=True)
```

```
data = api.measurements(city='Delhi',
parameter='pm25', limit=60000, df=True)
```

```
<response [200]>
averagingPeriod.unit averagingPeriod.value city country ... parameter sourceName unit value
lastUpdated
2021-04-20 23:30:00 seconds 3600.0 Delhi IN ... pm25 StateAir_NewDelhi b'\xc2\x5g/m\x2\x3' 78.00
2021-04-20 19:00:00 seconds 3600.0 Delhi IN ... co caaqm b'\xc2\x5g/m\x2\x3' 120.00
2021-04-20 19:00:00 seconds 3600.0 Delhi IN ... pm25 caaqm b'\xc2\x5g/m\x2\x3' 30.06
2021-04-20 19:00:00 seconds 3600.0 Delhi IN ... no2 caaqm b'\xc2\x5g/m\x2\x3' 16.29
2021-04-20 19:00:00 seconds 3600.0 Delhi IN ... o3 caaqm b'\xc2\x5g/m\x2\x3' 32.38
2021-04-20 19:00:00 seconds 3600.0 Delhi IN ... so2 caaqm b'\xc2\x5g/m\x2\x3' 8.26
2021-04-20 19:00:00 seconds 900.0 Delhi IN ... pm10 caaqm b'\xc2\x5g/m\x2\x3' 120.70
2021-04-20 19:00:00 seconds 3600.0 Delhi IN ... co caaqm b'\xc2\x5g/m\x2\x3' 1020.00
2021-04-20 19:00:00 seconds 3600.0 Delhi IN ... o3 caaqm b'\xc2\x5g/m\x2\x3' 44.60
2021-04-20 19:00:00 seconds 900.0 Delhi IN ... so2 caaqm b'\xc2\x5g/m\x2\x3' 6.20
2021-04-20 19:00:00 seconds 3600.0 Delhi IN ... no2 caaqm b'\xc2\x5g/m\x2\x3' 26.30
2021-04-20 19:00:00 seconds 3600.0 Delhi IN ... pm25 caaqm b'\xc2\x5g/m\x2\x3' 14.90
2021-04-20 19:00:00 seconds 3600.0 Delhi IN ... pm10 caaqm b'\xc2\x5g/m\x2\x3' 91.00
2021-04-20 19:00:00 seconds 3600.0 Delhi TN ... no2 caaqm b'\xc2\x5g/m\x2\x3' 16.27
```

PM_{2.5} over Delhi from OpenAQ Sensor Network

