Week 7 – JSON

JSON (JavaScript Object Notation), http://www.json.org, is the third of the main data interchange formats that we will look at. It is often touted as a "lightweight" format, in that it makes use of representations of data structures that are both easy for humans to read and for parsers to translate into internal data structures.

Now we'll make a collection to hold some earthquake data from the USGS earthquake website:
http://earthquake.usgs.gov/earthquakes/feed/v1.0/geojson.php

This page shows the format of the JSON that can be downloaded from this website. Let's use the "significant earthquakes" from the past 30 days.

```
>>> import urllib.request
>>> import json
>>> earthquake_url =
"http://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/significa
nt_month.geojson"
```

This gets the result from the website (which is in Python bytes) and converts it to a string using the decode() function.

```
>>> response = urllib.request.urlopen(earthquake_url)
>>> json_string = response.read().decode('utf-8')
```

Now we use the JSON package to transform the string to Python data structures consisting of lists and dictionaries. The outermost level is a dictionary, and we can look at the keys, comparing them with the format displayed at the website.

```
>>> eq_parsed_json = json.loads(json_string)
>>> type(eq_parsed_json)
<class 'dict'>
>>> eq_parsed_json.keys()
dict_keys(['metadata', 'features', 'type', 'bbox'])
>>> eq_parsed_json['type']
'FeatureCollection'
>>> eq_parsed_json['metadata']
{'url':
'https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/significant_month.geojson', 'title': 'USGS Significant Earthquakes, Past Month',
'status': 200, 'count': 9, 'api': '1.5.6', 'generated': 1489028073000}
```

We can dive deeper into the nested dictionaries:

```
>>> title = eq_parsed_json['metadata']['title']
>>> title
'USGS Significant Earthquakes, Past Month'
```

Now the earthquakes themselves are in a list under ‘features’. Let’s get the first one and look at its structure, again comparing with the website:

```
>>> quakelist = eq_parsed_json['features']
>>> len(quakelist)
9


>>> quake1 = quakelist[0]
>>> type(quake1)
<class 'dict'>
>>> quake1.keys()
dict_keys(['geometry', 'id', 'properties', 'type'])
```

We can continue to dive deeper into the structure of the data, but we can also get a good view of the format of a quake by invoking the pretty print function.

```
>>> print(json.dumps(quake1, indent=2))
{
  "properties": {
    "sig": 613,
    "felt": 3,
    "ids": ",us1000876f,pt17064051,at00omd5zx,",
    "updated": 1488816451040,
```

"status": "reviewed",
"alert": "green",
"rms": 0.99,
"tz": 600,
"net": "us",
"type": "earthquake",
"dmin": 3.326,
"sources": ",us,pt,at,",
"nst": null,
"magType": "mww",
"mmi": 6.06,
"cdi": 8.6,
"code": "1000876f",
"mag": 6.3,
"url": "https://earthquake.usgs.gov/earthquakes/eventpage/us1000876f",
"title": "M 6.3 - 32km NW of Kandrian, Papua New Guinea",
"tsunami": 1,
"gap": 17,
"place": "32km NW of Kandrian, Papua New Guinea",
"types": ",dyfi,geoserve,impact-link,losspager,moment-tensor,origin,phase-data,shakemap,",

```
    "detail":
"https://earthquake.usgs.gov/earthquakes/feed/v1.0/detail/us1000876
f.geojson",
    "time": 1488754073310
  },
  "geometry": {
    "type": "Point",
    "coordinates": [
      149.353,
      -5.999,
      31.07
    ]
  },
  "type": "Feature",
  "id": "us1000876f"
}
```