Data structures and formats exercise

This exercise aims to let you discover three different data structures available on the web. These structures vary significantly in what data they can represent and the ability to navigate and manipulate the data easily.

1) Tabular data

The most common data structure for open data is tabular.

Here, data is organised into rows and columns listing sequential values, such as expenditure.

If the data is record based and the relationships between records are not important, then tabular is an ideal structure.

Exercise

Take a look at the following tabular dataset available from the Foreign and Commonwealth Office:

http://data.gov.uk/dataset/financial-transactions-data-fco

Q1: How do you get hold of the data?

Q2: What capabilities does having data in this format provide?

Q3: What capabilities are missing when data is in this structure and format?

Developing applications with tabular data

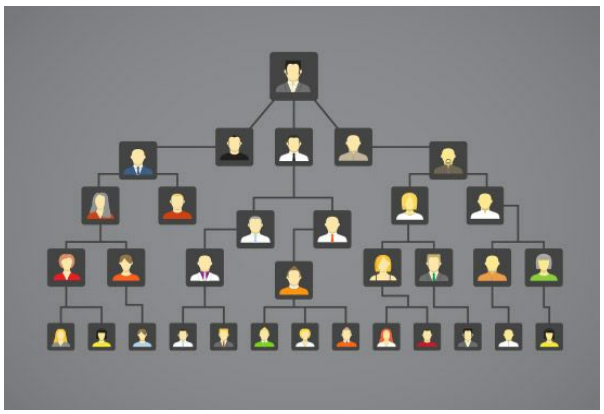
Tabular data comes in rows and columns. When developing applications, the *array* or *list* structure is the most common way to represent tabular data. Both are an essential set of items that can be referenced by row or item number (just like a table) and by key (column title).

For example if our table has **four** rows and **four** columns (name, address, postcode, phone number) to get the **address** of the **second** person you would need to reference it in your application as:

```
array[1]["address"]
```

(Note: In most programming languages. Also computers start counting from 0 not 1, so 1 is the second element of the array)

2) Hierarchical data



Hierarchical data shows the relationships between data points, such as a family tree or counties in each country.

Here the relationships between data points are of critical importance, however such relationships can only exist as parent-child relationships in a hierarchical structure.

Exercise

Take a look at a the following hierarchical dataset available from the the UK Government:

https://www.gov.uk/trade-tariff/sections

Q1: How do you get hold of the data?

A1: You need to add a ".json" to the end of the address in your browser. To view the data you might want to simply put the URL (including the ".json" extension) into http://jsonlint.com/

Q2: What capabilities does having data in this format provide?

Why not try using http://konklone.io/json/ to convert the data from JSON to CSV?

Q3: What capabilities are missing when data is in this structure and format?

Developing applications with hierarchical data

While a single axis of hierarchical data could be represented as a table (e.g. the list of all my grandparents), it is the relationships that are important, not the data. This means that it is often the navigation methods that are important.

Hierarchical data always operates using parent-child relationships, so to navigate between elements of a hierarchy the code would look like the following (node is where I currently am).

```
node.getParent().getParent(); (javascript)
```

Using a web browser, what is the grandparent of the following URL?

https://www.gov.uk/trade-tariff/headings/5703

3) Network data

Network structured data allows relationships to exist between any combination of elements in any direction.

A good example of a network data structure are social networks. Think of your friends and their friends on facebook or 1st, 2nd and 3rd degree contacts on Linkedin.

The web itself is a network, where pages can link to any number of other pages in any direction.

Exercise

Take a look at the following network dataset available from wikipedia:

http://dbpedia.org/resource/One_Direction

Q1: How do you get hold of the data?

A1: If you have opened it in a web page, you can scroll to the bottom of the page to get the data.

Additionally why not try to open the above URI in the data browser available from the University of Southampton at http://graphite.ecs.soton.ac.uk/browser/

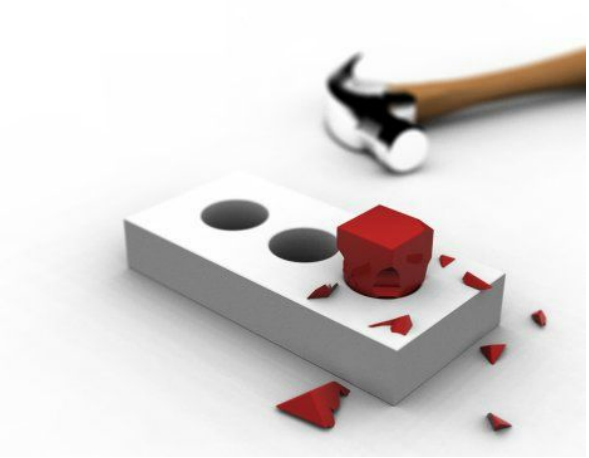
Q2: What capabilities does having data in this format provide?

Q3: What capabilities are missing when data is in this structure and format?

Developing applications with network data

Developing applications with network data is pretty much the same as with hierarchical data except you have to realise that you might end up going round in circles, as a node's grandparent could be the node itself. This makes it a challenge to recursively iterate over all elements without going round in circles and getting stuck.

Thinking structures and formats



During the course of this exercise you have used data available in the Comma separated values (CSV) and JavaScript Object Notation (JSON) formats as well as that serialised according to the Resource Description Framework (RDF) specification (which is not a format).

Each provides different capabilities and features.

Exercise

Q1: Does each format represent the same type of data? If not, what are the different types?

Q2: Which is the easiest to use?

Q3: What features of each format would be nice to see in others and why?