Session 3.
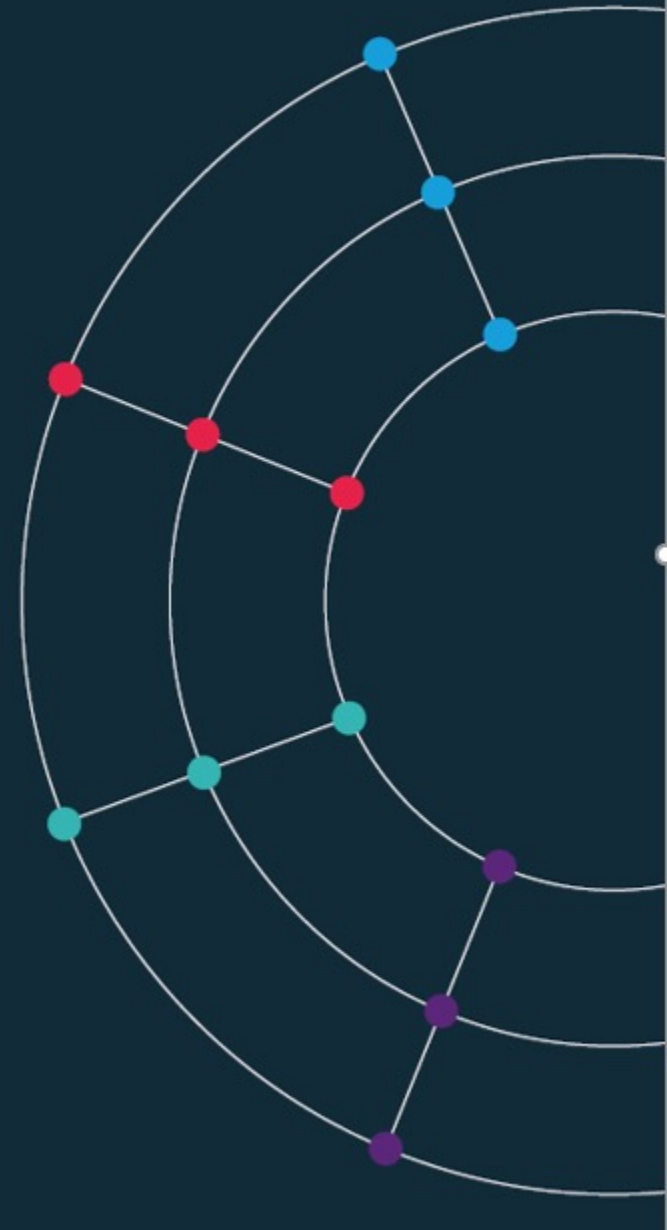
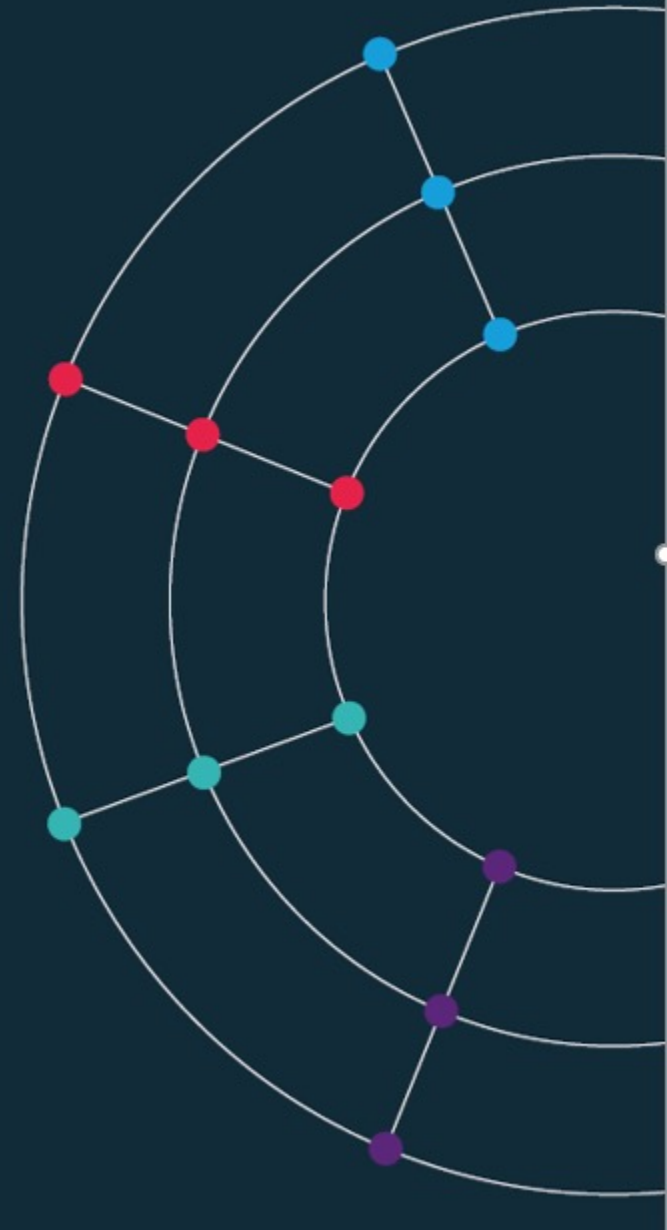*Accessing data programmatically*

Session 3.

Accessing data programmatically

Introduction to control functions

The idea.

We want our programs/analysis to take decisions for us. Not to continue doing the same thing again and again, but to be able to decide what to do next.

Without control structures (AKA 'flow control') programs don't do much.

What might you want a program to do for you?

- Stop or start.

- Take a decision on what to do next. Do different things in different conditions:
  - Time of day, or days of the week;
  - If data has certain properties: (stock market alert).

- Do something many times.
  - Dynamic programming / maximisation;
  - Batches of analysis: downloading, cleaning, charting.

The big three.

Sequence | Iteration | Conditionality/Selection

Programming languages typically feature three types of control:

- Sequence. Tells the program the order to do things in.

- Selection/conditionality. Makes a decision based on a testable condition.

- Iteration. Repeats a command over and over again, until a condition is met. Then stop and continue the code.

Selection   Iteration   Sequence
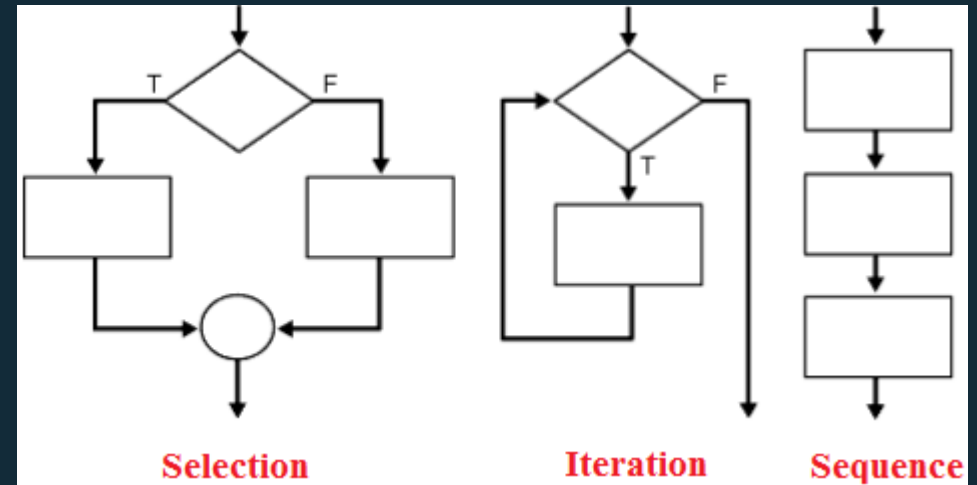
Control in data science.

If-else | Loops

In practical terms during a career in data you are going to make daily use two particular examples of these general ideas.

- If-else. Test some condition in your data. Based on the results of this test, take a number of different actions.

- Loops. Do the same thing to many pieces of data, many variables, many data sets. Or do the same thing on a number of different days.

These can be combined all possible ways.
- An if statement inside an if (AKA "nested")
- A loop withing a loop.
- If inside a loop
- Loop inside an if.

How many languages❓

Five?

How many languages?

Three?

Loops.
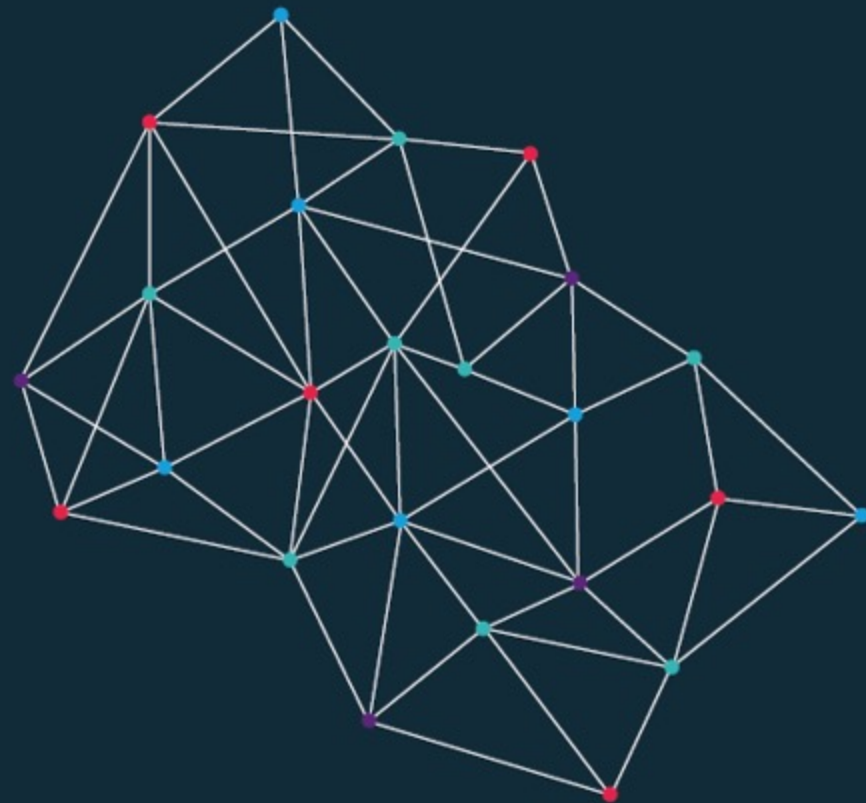
Loops.

```
// LOOPS

// For loops - two types:

forvalues i=1/100{
    display `i'
}

forvalues i=1(10)100{
    display `i'
}

// Generating ratios of variables to GDP
foreach i in debt deficit currentAccount investment consumption{
    gen `i'_ratio = `i'/gdp
}
```

Loops.

```
for i in range (1, 100):
    print i

for i in range (1, 10, 100):
    print i

for i in range (1, 100, 10):
    print i


variables = ["debt", "deficit", "GDP", "inflation"]
for i in variables:
  print(i)
```

An example of subtle differences between languages / functions.

Loops.

```
for (statement_1; statement_2; statement_3) {
    // Code block to run
}
```

What happens here:

- Statement_1 runs once, before the code block starts.

- Statament_2 defines a condition that must hold for the code block to run.

- Statement_3 runs each time the code block has been executed.

Loops.

STATA | Python | JavaScript

```javascript
for (let i = 1; i < 101; i++) {
    console.log(i);
  }



for (let i = 1; i < 101; i+10) {
    console.log(i);
  }
```

Loops.

STATA | Python | JavaScript

```javascript
// Set a list of variables:
variables = ["debt", "deficit", "GDP", "inflation"]

// We can index these:
console.log(variables)
console.log(variables[0])
console.log(variables[3])

// Work out how long this thing is:
len = variables.length

// Iterate though it, printing out each particular variable
for (let i=0; i<len; i++) {
    x = variables[i]
    console.log(x)
}
```

APIs.
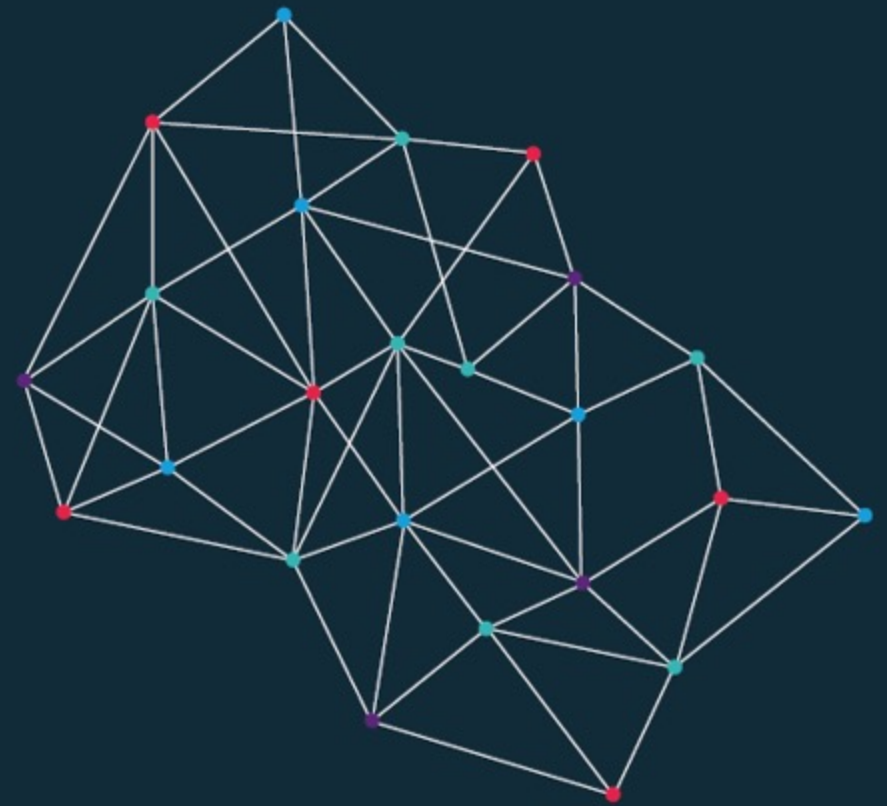
What is an API?

- Application Programming Interface

- An API is a software intermediary that allows two applications to talk to each other.

- They are everywhere: each time you use an app like Facebook or Instagram, send an instant message, or check your weather app on your phone, you are using an API (example: Apple Watch)

- APIs are extremely useful to data scientists because they provide a way to share/access data

API guidance.

- They all look different but have a similar set up.

- A base url: e.g. **https://api.stlouisfed.org/fred/series/observations?**

- A series of options you can choose: series_id= file_type= time_start=

- Often a request for your API key: api_key=

- Often, when the API requires more information/choices from you, a series of &
  symbols. An example:

    https://api.stlouisfed.org/fred/series/observations?series_id=UNRATE&api_key=22ee7a76e736e32f54f5
    df0a7171538d&file_type=json

api.stlouisfed.org/fred/series/observations?series_id=PCEPI&api_key=22ee7a76e736e32f54f5df0a7171538d&file_type=json

{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","observation_start":"1600-01-01","observation_end":"9999-12-31","units":"lin","output_type":1,"file_type":"json","order_by":"observation_date","sort_order":"asc","count":752,"offset":0,"limit":1...01","value":"16.042"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date...10-14","realtime_end":"2021-10-14","date":"1959-04-01","value":"16.1"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","dat...06-01","value":"16.155"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1959-07-01","value":"16.189"},{"realtime_s...{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1959-09-01","value":"16.255"},{"realtime_start":"2021-10-14","realt...14","realtime_end":"2021-10-14","date":"1959-11-01","value":"16.304"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date...01-01","value":"16.314"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1960-02-01","value":"16.331"},{"realtim...{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1960-04-01","value":"16.4"},{"realtime_start":"2021-10-14","realtim...14","realtime_end":"2021-10-14","date":"1960-06-01","value":"16.424"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date...08-01","value":"16.481"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1960-09-01","value":"16.491"},{"realtime_s...{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1960-11-01","value":"16.565"},{"realtime_start":"2021-10-14","realt...14","realtime_end":"2021-10-14","date":"1961-01-01","value":"16.571"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date...03-01","value":"16.578"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1961-04-01","value":"16.568"},{"realtime_s...{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1961-06-01","value":"16.585"},{"realtime_start":"2021-10-14","realt...14","realtime_end":"2021-10-14","date":"1961-08-01","value":"16.635"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date...10-01","value":"16.652"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1961-11-01","value":"16.653"},{"realtime_s...{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1962-01-01","value":"16.689"},{"realtime_start":"2021-10-14","realt...14","realtime_end":"2021-10-14","date":"1962-03-01","value":"16.756"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date...05-01","value":"16.786"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1962-06-01","value":"16.796"},{"realtime_...{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1962-08-01","value":"16.811"},{"realtime_start":"2021-10-14","realt...14","realtime_end":"2021-10-14","date":"1962-10-01","value":"16.876"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date...12-01","value":"16.882"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1963-01-01","value":"16.923"},{"realtime_s...{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1963-03-01","value":"16.928"},{"realtime_start":"2021-10-14","realt...14","realtime_end":"2021-10-14","date":"1963-05-01","value":"16.954"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date...07-01","value":"17.025"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1963-08-01","value":"17.048"},{"realtime_s...{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1963-10-01","value":"17.078"},{"realtime_start":"2021-10-14","realt...14","realtime_end":"2021-10-14","date":"1963-12-01","value":"17.127"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date...02-01","value":"17.188"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1964-03-01","value":"17.198"},{"realtime_...{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1964-05-01","value":"17.213"},{"realtime_start":"2021-10-14","realt...14","realtime_end":"2021-10-14","date":"1964-07-01","value":"17.259"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date...09-01","value":"17.299"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1964-10-01","value":"17.31"},{"realtime_st...{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1964-12-01","value":"17.359"},{"realtime_start":"2021-10-14","realt...14","realtime_end":"2021-10-14","date":"1965-02-01","value":"17.385"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date...04-01","value":"17.435"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1965-05-01","value":"17.474"},{"realtime_s...{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1965-07-01","value":"17.538"},{"realtime_start":"2021-10-14","realt...14","realtime_end":"2021-10-14","date":"1965-09-01","value":"17.55"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date"...01","value":"17.585"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1965-12-01","value":"17.649"},{"realtime_star...10-14","realtime_end":"2021-10-14","date":"1966-02-01","value":"17.743"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","d...14","date":"1966-04-01","value":"17.848"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1966-05-01","value":"17.8...{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1966-07-01","value":"17.949"},{"realtime_start":"2021-10-14","realt...14","realtime_end":"2021-10-14","date":"1966-09-01","value":"18.075"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date...11-01","value":"18.15"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1966-12-01","value":"18.187"},{"realtime_st...{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1967-02-01","value":"18.209"},{"realtime_start":"2021-10-14","realt...14","realtime_end":"2021-10-14","date":"1967-04-01","value":"18.249"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date...06-01","value":"18.343"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1967-07-01","value":"18.405"},{"realtime_s...{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1967-09-01","value":"18.519"},{"realtime_start":"2021-10-14","realt...14","realtime_end":"2021-10-14","date":"1967-11-01","value":"18.632"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date...01-01","value":"18.748"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1968-02-01","value":"18.824"},{"realtime_s...{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1968-04-01","value":"18.945"},{"realtime_start":"2021-10-14","realt...14","realtime_end":"2021-10-14","date":"1968-06-01","value":"19.075"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date...08-01","value":"19.212"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1968-09-01","value":"19.279"},{"realtime_s...{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date":"1968-11-01","value":"19.43"},{"realtime_start":"2021-10-14","realti...14","realtime_end":"2021-10-14","date":"1969-01-01","value":"19.546"},{"realtime_start":"2021-10-14","realtime_end":"2021-10-14","date...

Raw JSON.

Download a JSON formatter plug in for your browser

JSON Formatter 0.6.0
Makes JSON easy to read. Open source.

```json
{
    "realtime_start": "2021-10-14",
    "realtime_end": "2021-10-14",
    "observation_start": "1600-01-01",
    "observation_end": "9999-12-31",
    "units": "lin",
    "output_type": 1,
    "file_type": "json",
    "order_by": "observation_date",
    "sort_order": "asc",
    "count": 885,
    "offset": 0,
    "limit": 100000,
    "observations": [
        {
            "realtime_start": "2021-10-14",
            "realtime_end": "2021-10-14",
            "date": "1948-01-01",
            "value": "3.4"
        },
        {
            "realtime_start": "2021-10-14",
            "realtime_end": "2021-10-14",
            "date": "1948-02-01",
            "value": "3.8"
        },
        {
            "realtime_start": "2021-10-14",
            "realtime_end": "2021-10-14",
            "date": "1948-03-01",
            "value": "4.0"
        },
        {
            "realtime_start": "2021-10-14",
            "realtime_end": "2021-10-14",
            "date": "1948-04-01",
            "value": "3.9"
        },
        {
            "realtime_start": "2021-10-14",
            "realtime_end": "2021-10-14",
            "date": "1948-05-01",
            "value": "3.5"
        },
        {
            "realtime_start": "2021-10-14",
            "realtime_end": "2021-10-14",
            "date": "1948-06-01",
            "value": "3.6"
```
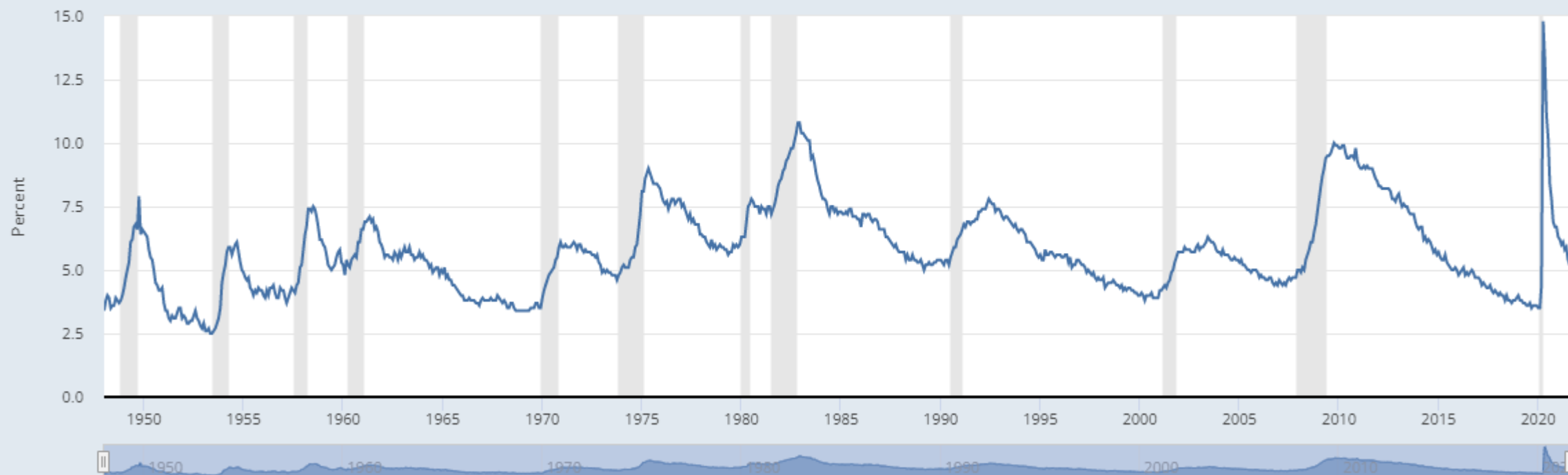
Worked example.

This chart pulls data from the Economics Observatory API:

```
 1   {
 2     "$schema": "https://vega.github.io/schema/vega-lite/v5.json",
 3
 4     "data": {"url": "https://api.economicsobservatory.com/gbr/unem?vega"},
 5
 6     "mark": "line",
 7
 8     "encoding": {
 9
10       "x": {"field": "date", "type": "temporal"},
11
12       "y": {"field": "value", "type": "quantitative"}
13     }
14   }
```
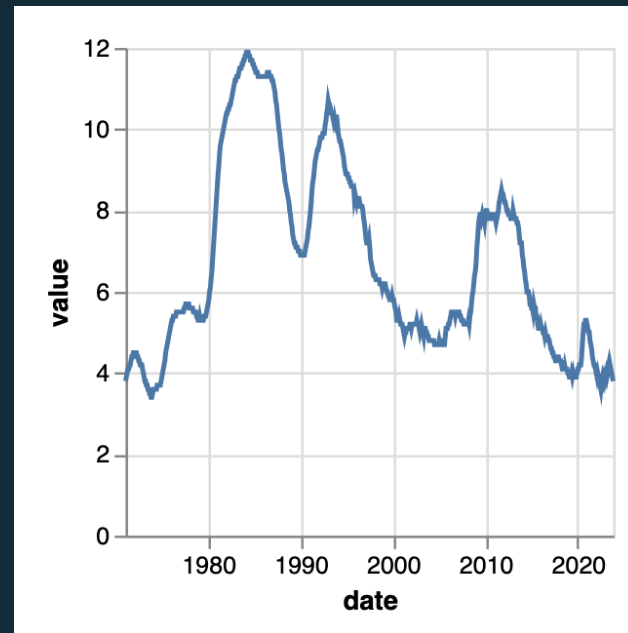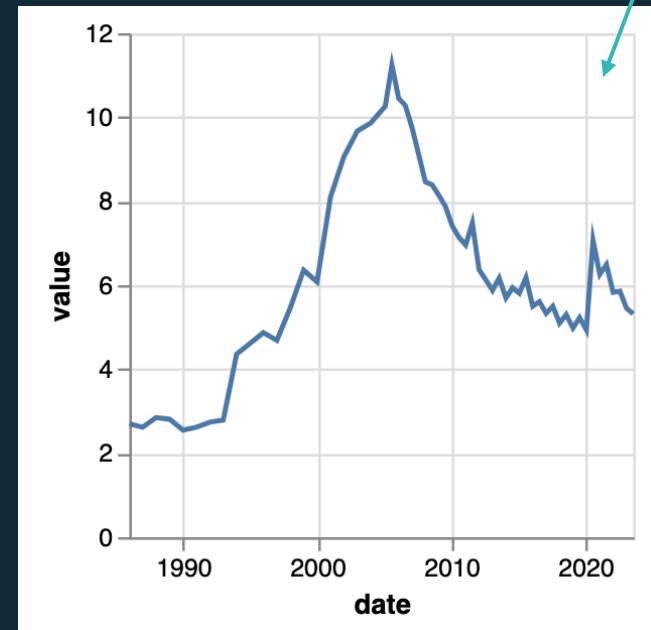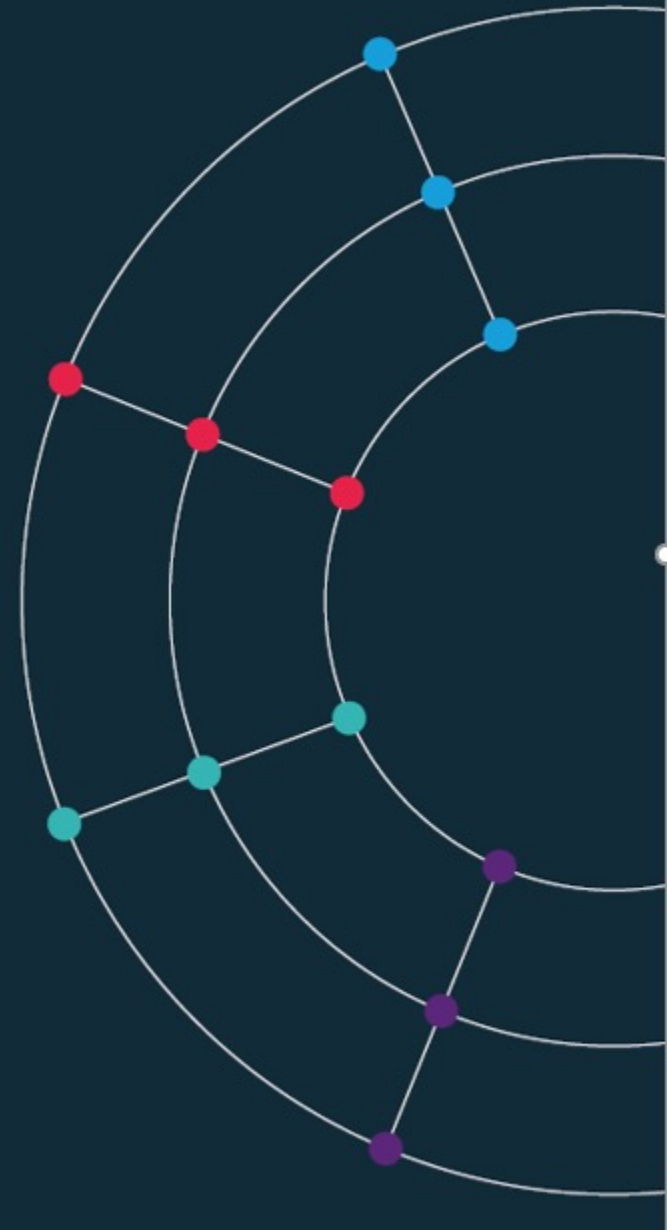
Worked example.

Edit the URL to draw data from a different country:

API URL tweaked

Chart now shows Indonesian data

```
1   {
2     "$schema": "https://vega.github.io/schema/vega-lite/v5.json",
3
4     "data": {"url": "https://api.economicsobservatory.com/idn/unem?vega"},
5
6     "mark": "line",
7
8     "encoding": {
9
10      "x": {"field": "date", "type": "temporal"},
11
12      "y": {"field": "value", "type": "quantitative"}
13    }
14  }
```

Session 3.

Accessing data programmatically

https://github.com/EconomicsObservatory/courses
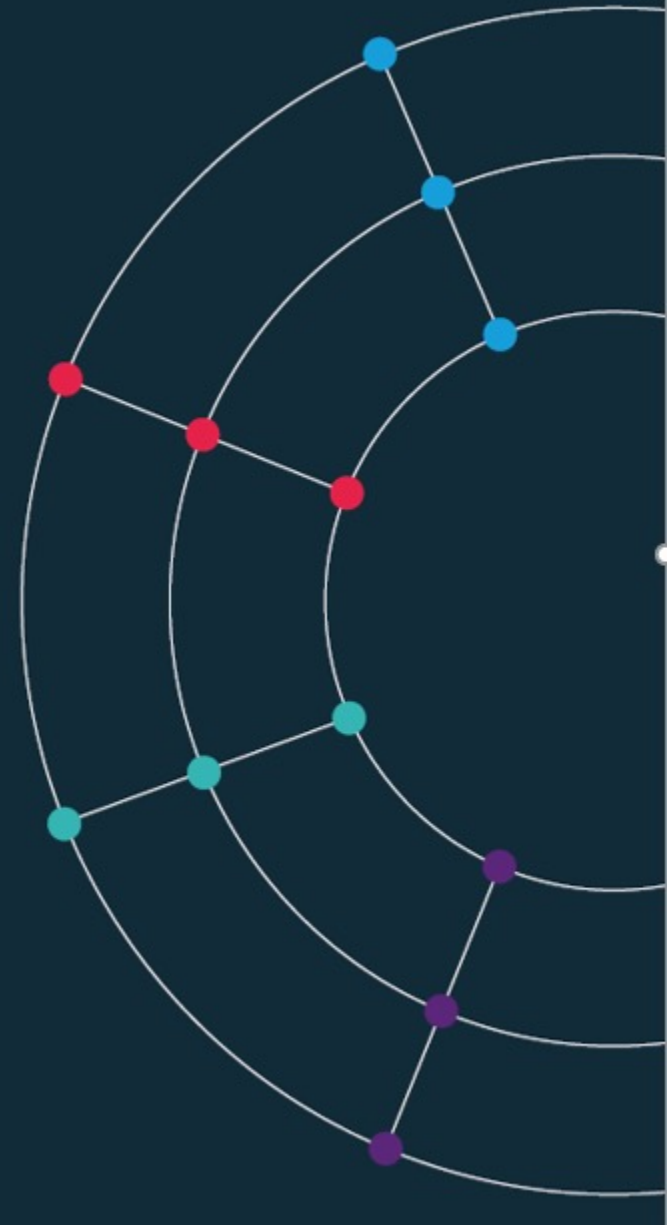
Code-along.

In this third practical session, we will be using Google Colab

1. Quick introduction to loops.

2. Using a loop with an API.

3. Use Python to batch download data

4. Embedding a chart using data accessed via batch download