APIs & Plumber R library

Dr Olly Butters
Public Health, Policy and Systems
olly.butters@liverpool.ac.uk

Learning objectives

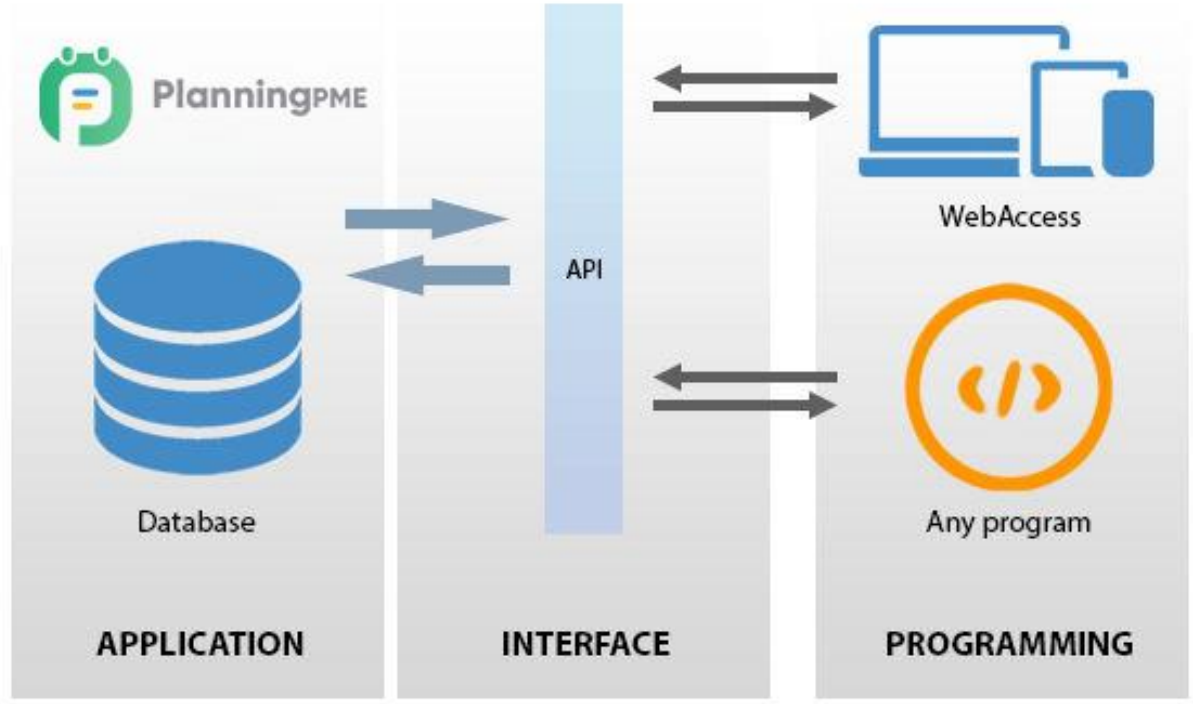
- Understand what an API is.
- Use existing APIs.
- Develop our own APIs.

Session overview

- What is an API, where are they used? (~25 mins)
- Exercise - Two R examples of using existing external APIs. (~15 mins)
- Introduction to plumber. (~15 mins)
- Exercise - Write some plumber code. (~30 mins)
- Final comments. (~5 mins)

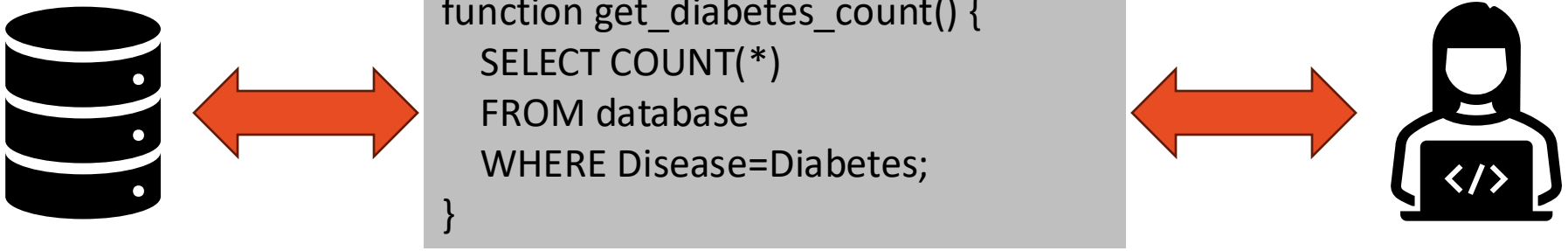
What is an API?

- Application Programming Interface.



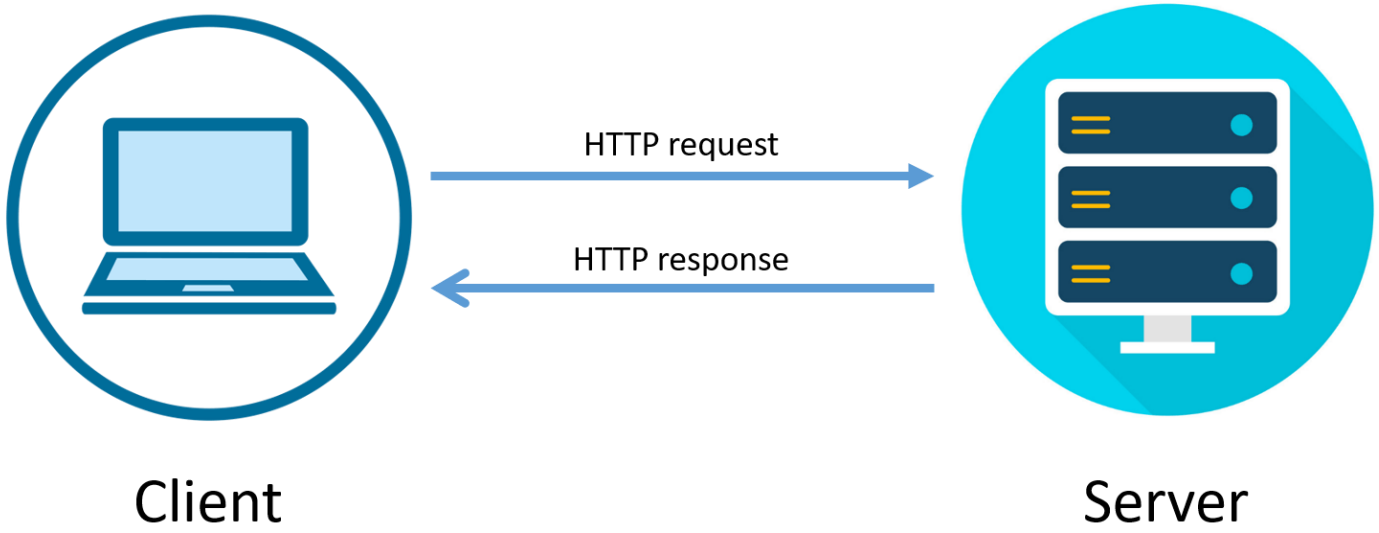
https://www.planningpme.com/planningpme-api.htm

Steps towards an API



Name	ID	Disease
John	2345	Diabetes
Jane	7853	Asthma
Judy	1337	Diabetes
Joe	8867	COPD

API messaging

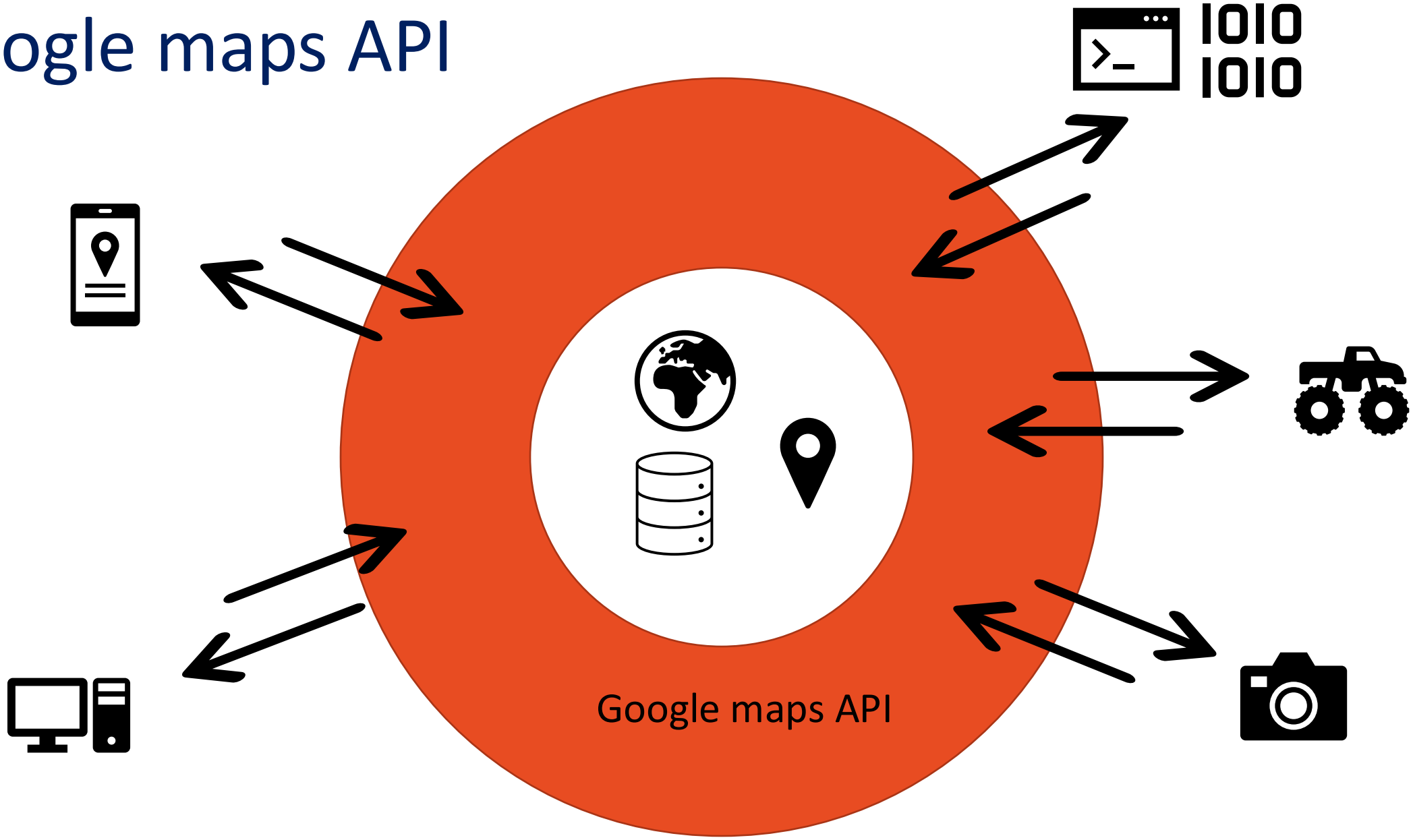


https://bytesofgigabytes.com/networking/how-http-request-and-response-works/

Why bother?

- Easier than connecting to underlying applications.
- Can make subset of data/application available.
- Common language.
- Static interface.

Google maps API



Example APIs

Get data

- Google maps
- Wikidata
- Fitbit
- British library
- Data.parliament.uk

Add data

- Twitter
- Facebook
- Instagram

Book appointments

- GP systems
- Restaurants

Manage services

- Amazon Web Services
- Microsoft Azure

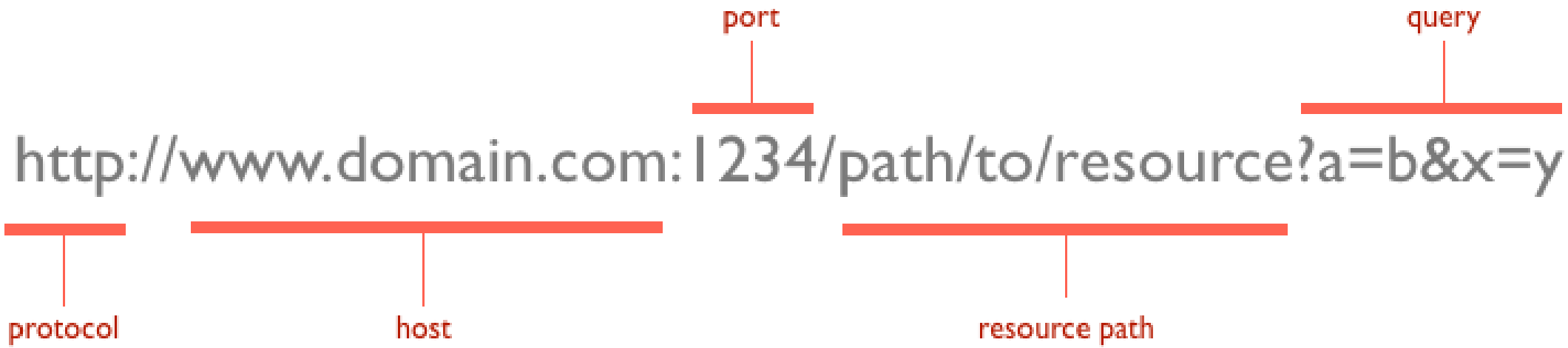
Real world interaction

- Google Nest
- Dishwasher
- Burglar alarm

Health data APIs

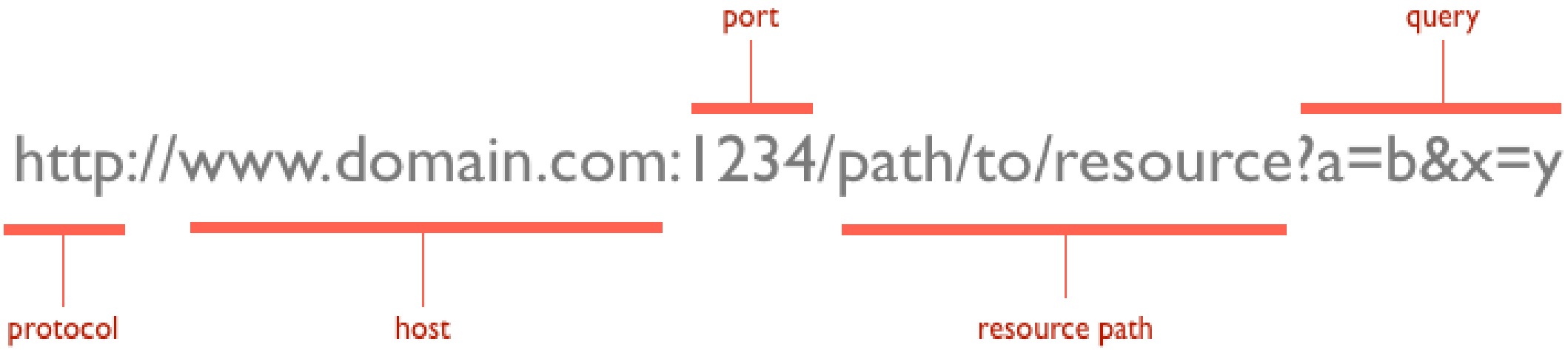
- Bioportal -> Look up ontologies etc.
- Government health stats -> Going to use these today
- NHS Digital -> Loads of APIs to find/get/add data
- Air quality.
- UK government API catalogue
- Urban Observatory
- https://data.police.uk/docs/
- Care Quality Commission -> locations of care homes

Anatomy of an API URL (the request)



Mongolariun, CC BY-SA 4.0 <https://creativecommons.org/licenses/by-sa/4.0>, via Wikimedia Commons

Anatomy of an API URL (the request)



Pavement	Liver building	Flat number 3	Kitchen/cupboard/get_mug	colour=red
Road	Liver building	Garage number 2	Park/car	

Query vs Path based API

Query based:

someapi.com:1234/flu_cases?location=england&type=hospitalisation

Path based:

someapi.com:1234/location/england/type/hospitalisation/flu_cases

Response

- Typically get a header and content in the response
- Response codes in header
  - 200 – OK
  - 404 – Not found
  - 500 – Internal server error
- Content is usually JSON or XML

Google maps elevation API call example

Request: https://maps.googleapis.com/maps/api/elevation/json?locations=39.7391536%2C-104.9847034

Response:


```
{
  "results":
    [
      {
        "elevation": 1608.637939453125,
        "location": { "lat": 39.7391536, "lng": -104.9847034 },
        "resolution": 4.771975994110107,
      },
    ],
  "status": "OK",
}
```

Exercises 1: Find out a useless fact

1. https://github.com/OllyButters/HDS-plumber/archive/refs/heads/main.zip
2. Open README.md
3. Open exercise_1_api_useless_facts.R
4. Run the R file one line at a time (Ctrl-Enter) and read the comments as you go.

Exercise 2: Write an R script to find out what the influenza hospitalisation rate in England is and plot it

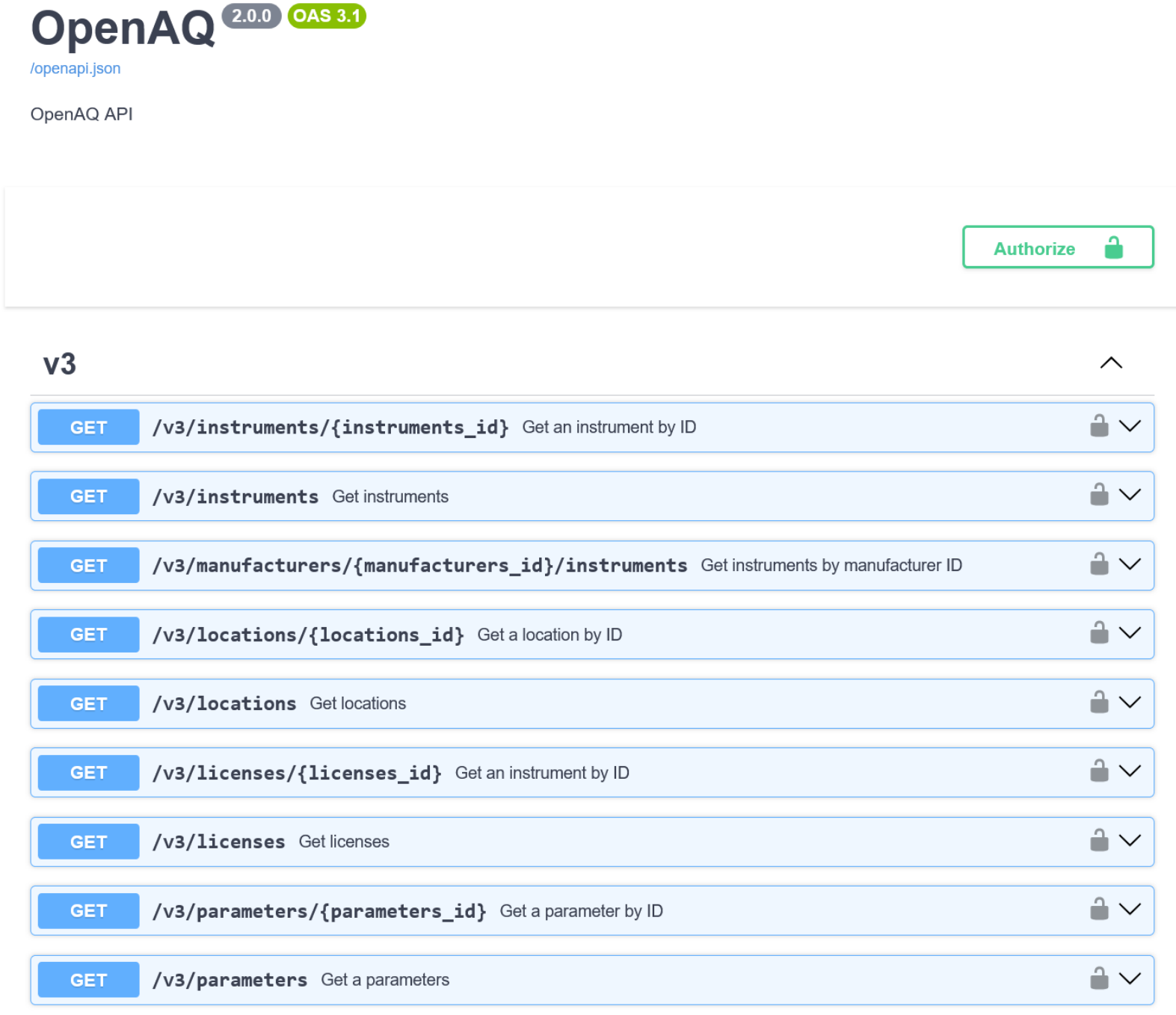
1. Start a new R file called exercise_2_api_influenza_rate.R
2. Copy the relevant parts from the first exercise.
3. The query URL (the request) is: https://api.ukhsa-dashboard.data.gov.uk/themes/infectious_disease/sub_themes/respiratory/topics/Influenza/geography_types/Nation/geographies/England/metrics/influenza_healthcare_hospitalAdmissionRateByWeek?age=all&page=2&page_size=365
4. Can copy URL from the README.md file.
5. Get the data from the API and plot it.
6. If you get really stuck you can look at exercise_2_influenza_answer.R

Exercise 1 & 2 summary

- Used two APIs to get data from remote services.
- Now we are going to build our own APIs and connect to them in a similar way.

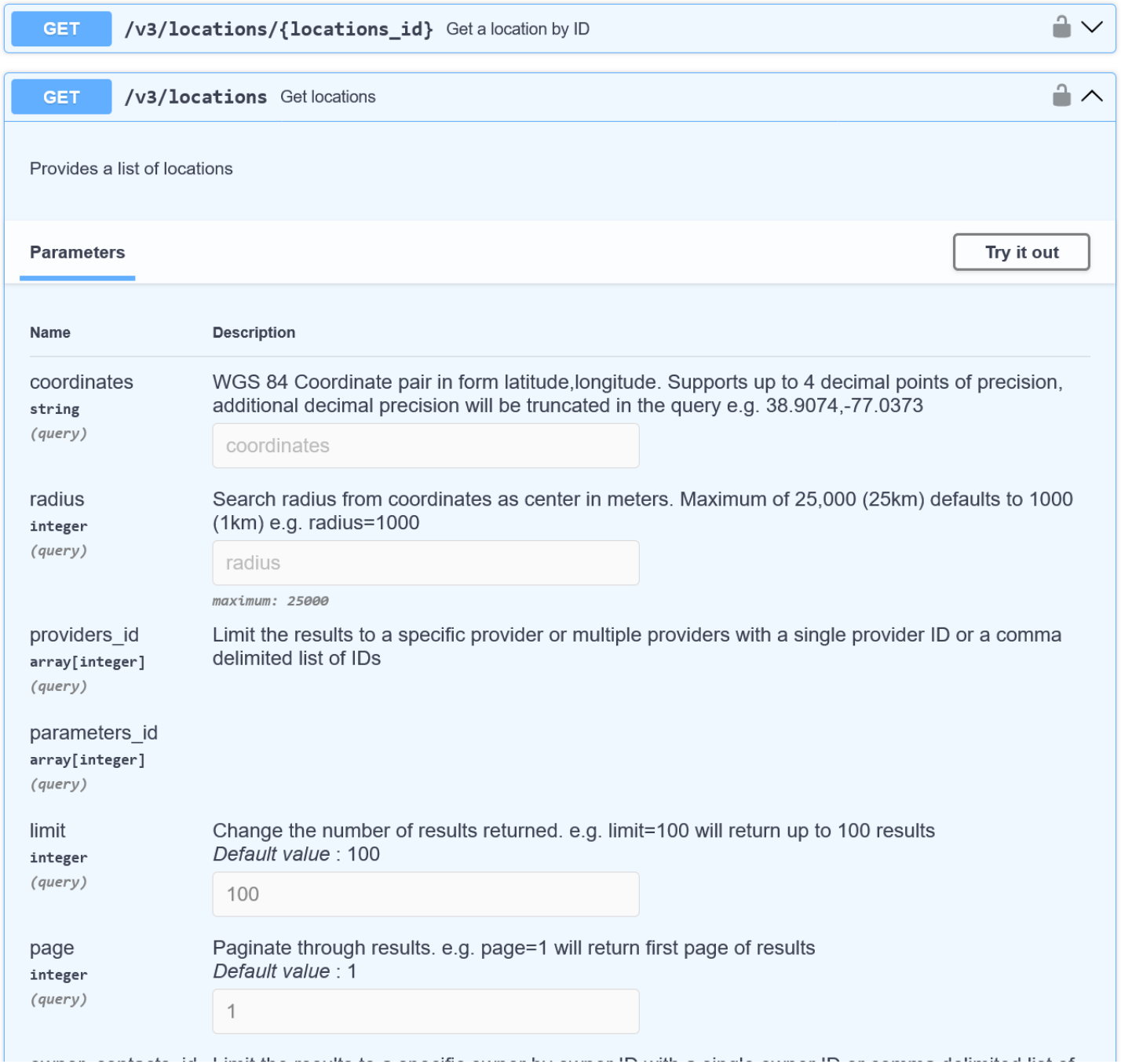
Swagger

- Web tool to help explore and use compliant APIs
- https://api.openaq.org/docs



Swagger

User interface for adding query parameters



Decorators

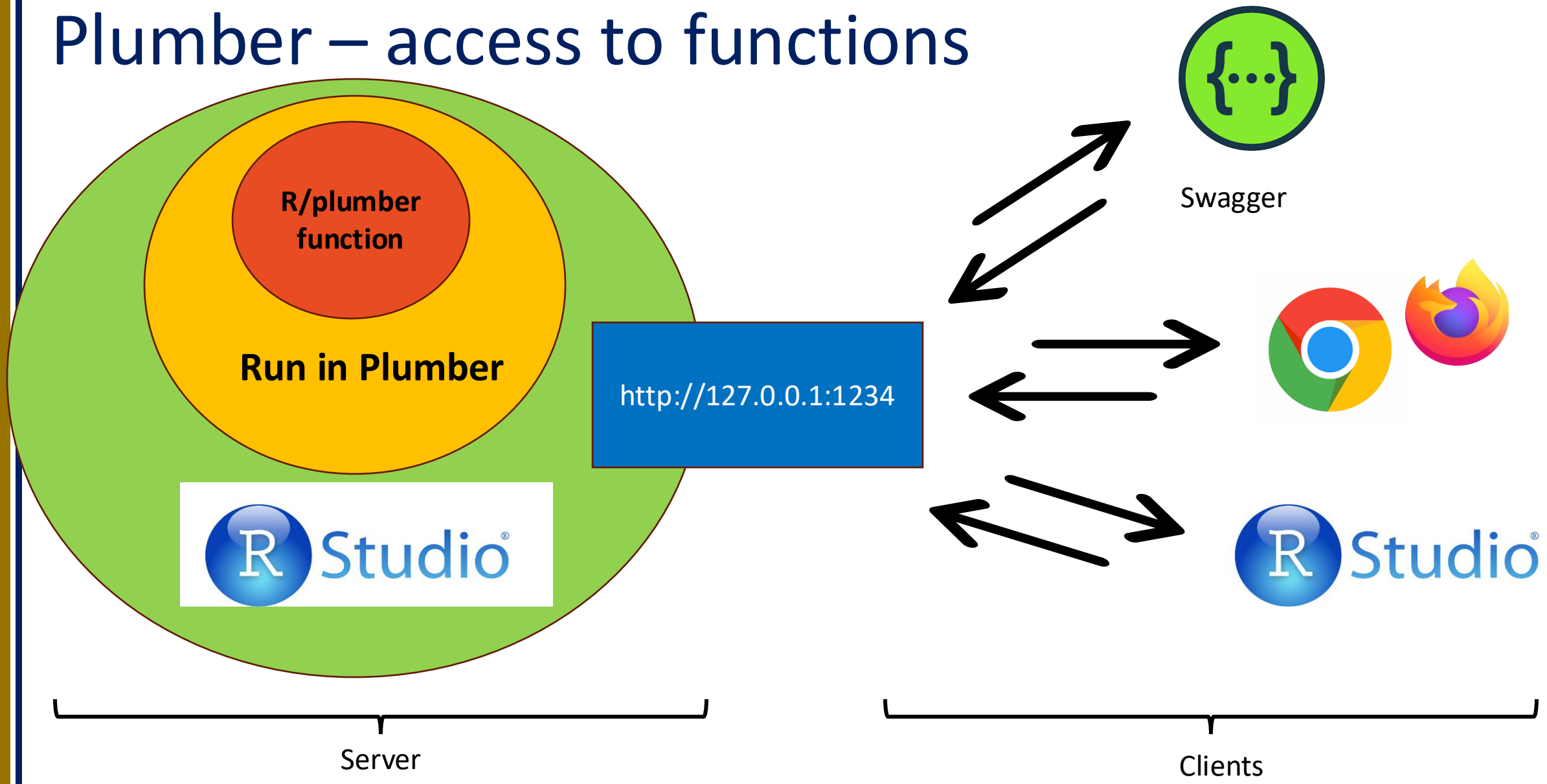
- Decorators let you modify function behaviour without modifying the function code!
- Start with a # so ignored most of the time.
- Common in other languages.

```
#* I am a decorator
my_function <- function()
{
   #do awesome stuff
}
```

Plumber function

```
#* Return the square of a number
#* @param a The number to square
#* @get /square
function(a) {
   as.numeric(a) * as.numeric(a)
}
```

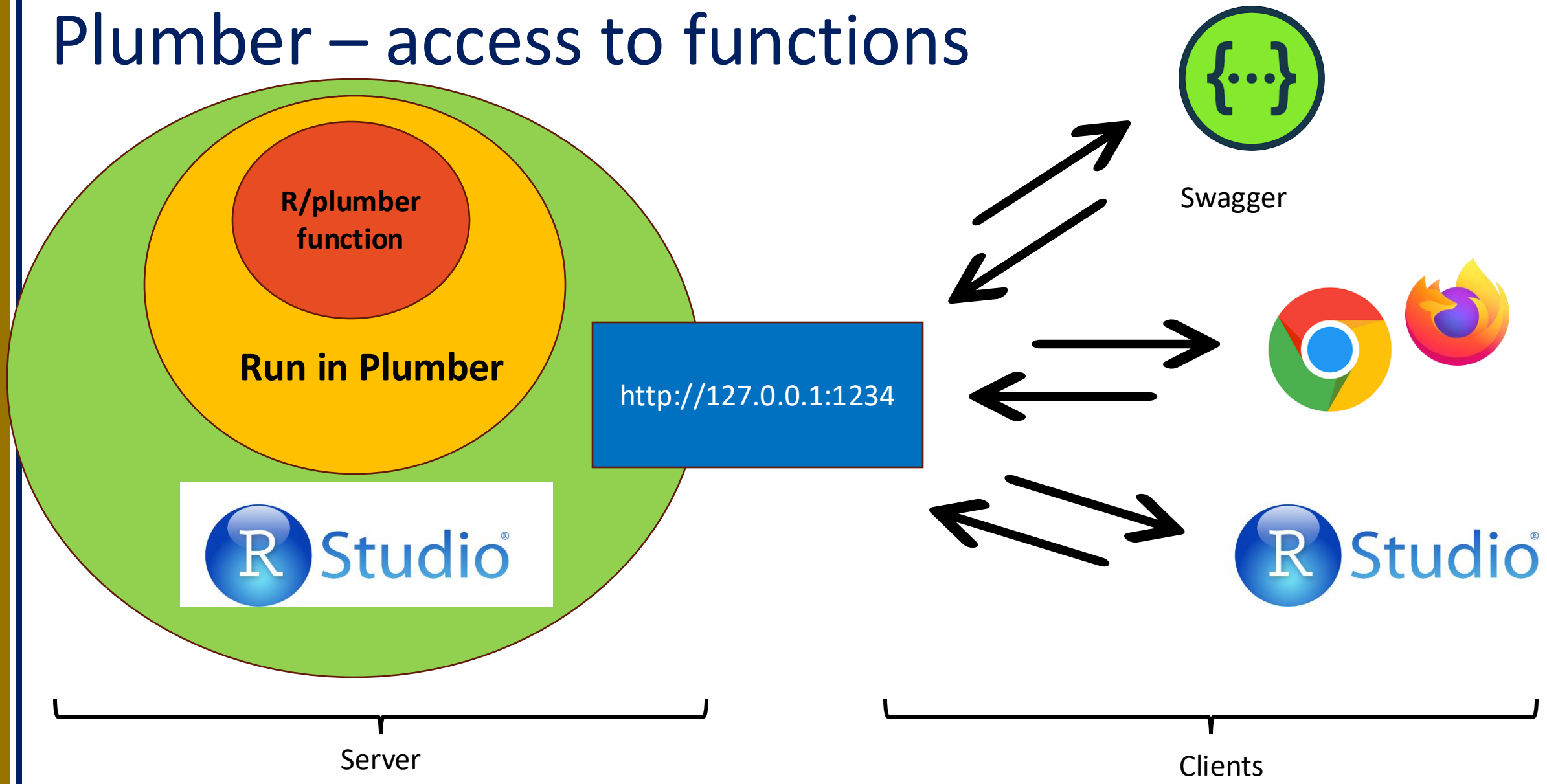
Plumber – access to functions



Exercise 3: Run some plumber code

- Open exercise_3_plumber_example_server.R
- Click on the "Run API" button on the top right of the code.
- This will open a web browser with swagger running in it
- Run example plumber functions (/hello, /square, /plot)
- Open request URLs directly in a web browser
- Open a second RStudio instance (Session > New Session), open exercise_3_plumber_example_client.R, run examples.
- More info in the README.md document

Plumber – access to functions



Exercises: write some plumber code

- Exercise 4: Write a plumber function to use gapminder data to show population of the UK in 1982. (Gapminder is a dataset of populations of various countries from 1952 - 2007).
- Exercise 5: Write a plumber function to allow a user to find out the population of any country during any year in gapminder.
- Exercise 6: Write a plumber function to plot the population change of a user defined country.

Additional points

- GET/POST
  - GET -> Usually GETting data, POST -> usually submitting data.
- Bounds checks -> What if I ask for gapminder data for this year?
- Security is vital on public APIs
- Good list of public APIs - https://github.com/public-apis/public-apis
- APIs can change or be retired!