3.3.3 Using Python requests library

As you've learnt so far this week, APIs rely on HTTP requests. Now, you'll learn more about how to write your own HTTP requests, using a Python library, called the Requests library. As you know, one of the advantages of Python is its modularity and the vast number of libraries available for you to use.

Although it isn't the only option, the Requests library is the go-to standard for making HTTP requests in Python. It hides the intricacies of making requests behind a simple and user-friendly API. In this topic, you will learn how to install the library, consider some advantages of using the Requests library, and, at the end, code a simple request.

Making GET requests

There are six main HTTP methods or verbs that you need to use as part of working with APIs, as follows: `GET`, `POST`, `HEAD`, `PUT`, `DELETE`, and `PATCH`. Of the six, `GET` and `POST` are the most frequently used, so this topic will focus on these two request methods. You'll look at the remaining four methods—`HEAD`, `PUT`, `DELETE`, and `PATCH`.

The primary function of the `GET` methods in APIs is to **retrieve** information. Let's say you want to pull the most recent information about the exchange rate for your local currency from an API. You would need to write a `GET` command in order to **get** that information. Since you will write more code, it's a good idea to organise it. You can organise existing code into a class, where you can store all the `GET` request logic that you will use.

To **retrieve** information from an API, you'll use functions from `urllib3`. The package consists of five modules, namely `request`, `response`, `error`, `parse`, and `robotparser`. You can read more about `urllib` in the official **Python documentation (https://docs.python.org/3/library/urllib.request.html#module-urllib.request)** (Python n.d.). Important to remember is that `requests` is a library in Python, while `urllib3` is a package. `urllib3` is mainly used to open APIs and return data in HTML format.

Websites can be written in HTML and JSON. The main difference is that **HTML** code has crocodile brackets (`<>`), while **JSON** has curly brackets (`{}`). Follow this **API link (https://web.archive.org/web/20210512072648/https://api.publicapis.org/entries)** and look at the information. What do you see? Do you think the code is in HTML or JSON?

Apply your learning

Head over to **Coding Rooms (https://app.codingrooms.com/app/org/swinburne-online-NHciF9n/course/cos60016-q1-programming-for-development-2023-piKGheK/b/3.3.2-http-requests-in-python-UHjRK6j)** to see how to get the status of a URL.