





Let there be Data!

APIs in Python

A Reproducible Research Workshop

Simon Stone

Research Data Services

Dartmouth College





About the Reproducible Research Group

- Joint venture of **Research Computing @ ITC** and **Research Data Services @ Library**
- Consult with **experts** on
 - research data management,
 - data visualization,
 - biomedical research support,
 - spatial data and GIS,
 - high performance and research computing,
 - statistical analysis,
 - economics and social sciences data
- **Meet** the people on campus that support your reproducible research lifecycle
- **Engage** in community discussions to learn from other researchers on campus
- Attend a workshop to **learn** practical tools and tips



About Research Data Services

Research Data Management

Data Management Plans (DMPs) for sponsored projects

Finding and using 3rd party data

Collection and cleaning of data

Organization and documentation

Publishing and Repositories

Data Analysis/Visualization

Textual, numeric, spatial data

Reproducible research workflows

Scripting in R: tidyverse core package (i.e. ggplot, dplyr, tydr, tibble, etc.)

Scripting in Python: NumPy, SciPy, Pandas, Scikit-learn, Matplotlib, Seaborn, (OpenCV, PyTorch, TensorFlow, Tesseract, NLTK, etc.)

Computational Scholarship

Computational project planning

Collections as Data

Storytelling with data and visualizations

Text and data mining

Digital Humanities support

Computational Pedagogy



Work with us

ResearchDataHelp@groups.dartmouth.edu

Jeremy Mikecz

Research Data Science Specialist
jeremy.m.mikecz@dartmouth.edu
dartgo.org/jeremyappts

Simon Stone

Research Data Science Specialist
simon.stone@dartmouth.edu
dartgo.org/meetwithsimon

Lora Leligdon

Head of Research Data Services
lora.c.leligdon@dartmouth.edu
dartgo.org/lora



What is an API?

👉 API stands for Application Programming Interface

What is an interface? (in software design)

- 🧩 A piece of software can be thought of in two parts:
 1. How it does what it does (the *implementation*)
 2. What the user of the software needs to put in and what they can get out (the *interface*)
- 🙌 The implementation rarely matters to a user
- 💯 The interface is all we need to know to use the software
- 👉 The implementation may change, as long as the interface stays the same
- 🤝 The interface is a contract between the program and the user

What is an interface? (in software design)

Example:

Interface

```
print(*objects, sep=' ', end='\n', file=None, flush=False)
```

Print *objects* to the text stream *file*, separated by *sep* and followed by *end*. *sep*, *end*, *file*, and *flush*, if present, must be given as keyword arguments.

All non-keyword arguments are converted to strings like `str()` does and written to the stream, separated by *sep* and followed by *end*. Both *sep* and *end* must be strings; they can also be `None`, which means to use the default values. If no *objects* are given, `print()` will just write *end*.

The *file* argument must be an object with a `write(string)` method; if it is not present or `None`, `sys.stdout` will be used. Since printed arguments are converted to text strings, `print()` cannot be used with binary mode file objects. For these, use `file.write(...)` instead.

Output buffering is usually determined by *file*. However, if *flush* is true, the stream is forcibly flushed.

Changed in version 3.3: Added the *flush* keyword argument.

Description of the interface

Notification of a change
in the “contract”





No imple-
mentation
details!

Source: <https://docs.python.org/3/library/functions.html#print>

What is an API?

- 👉 API stands for Application Programming Interface
- 🧱 It allows us to use someone else's “code blocks” to build applications/programs/scripts
- 🔧 Many APIs are used to bring in functionality (e.g., processing images)
- 📈 Here, we will focus on APIs that are used to bring in *data*!

Why use an API over the web?

-  An API lets us search, filter, query, and retrieve data using someone else's code
-  Such an API is usually used via the internet (a *Web API*):
 -  Only download the data you actually want
 -  Data is always current



What you will learn in this workshop

- **How** does software communicate via the web
- **What** is a RESTful web API
- **How** do you use a RESTful web API
- **What** is an API wrapper
- **How** using a Python wrapper can make your life easier

What we will work with in this workshop

- Platform: <https://jhub.Dartmouth.edu>
- **Python**
- Materials: www.dartgo.org/rr-apis-in-python





Let's get started...

RESTful communication on the internet

REpresentational State Transfer (REST)

A way to structure network-based applications

An application is *RESTful*, if:

- **Resources** are being transferred (e.g., a web page, images, list of names, sensor data, ...)
- The user interacts with the application using a **limited set of actions**
- The application is **stateless**



RESTful communication on the internet

REpresentational State Transfer (REST)

A way to structure network-based applications

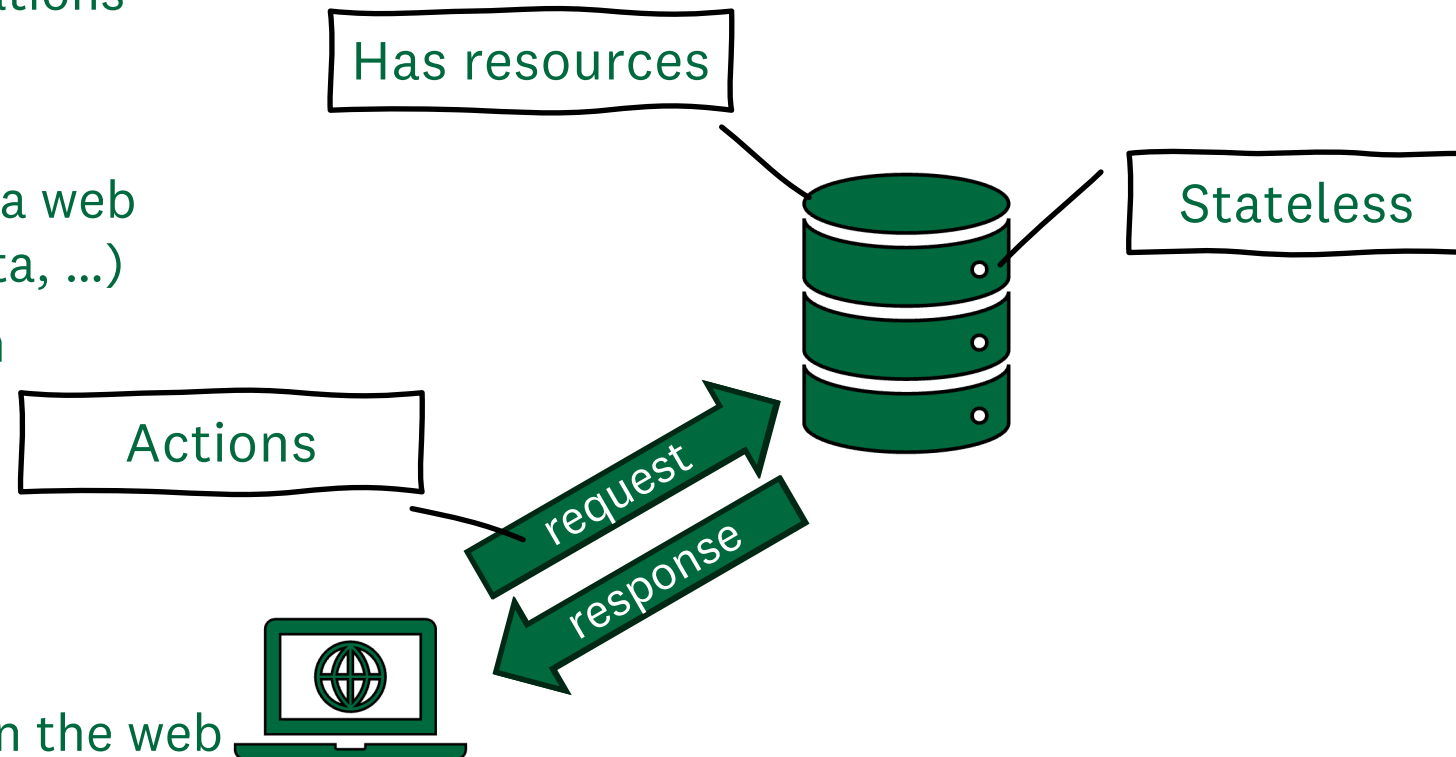
An application is *RESTful*, if:

- **Resources** are being transferred (e.g., a web page, images, list of names, sensor data, ...)
- The user interacts with the application using a **limited set of actions**
- The application is **stateless**

HyperText Transfer Protocol (HTTP)

The protocol governing communication on the web

👍 A (mostly) RESTful application





Let's see that in
action...



Takeaways

- APIs are a great way to automate data retrieval
- APIs allow specific and structured data access (cf. scraping)
- If a Python wrapper is available, we can write clear, concise, and reproducible code
- Whenever you need data from a source on the web:
 1. Check if it offers an API
 2. Check if there is a Python wrapper

Next steps

- Try things!

🪐 NASA: <https://api.nasa.gov/>

💬 ChatGPT: <https://github.com/openai/openai-python>

🎵 Spotify: <https://spotipy.readthedocs.io/>

🎓 Web of Science:
<https://researchguides.dartmouth.edu/c.php?g=59725&p=9910244>



Thank you.

