CODING NOMADS

9) HTTP & Web Services Lesson

RESTful APIs & Endpoints

7 min to complete · By Ryan Desmond, Jared Larsen

Contents

This section began by explaining web services and introducing you to RESTful services. But, it is necessary to understand the technologies these services are built upon before you actually start working with them.

After navigating the previous chapters, you should now have some understanding of both the HTTP protocol and the JSON data format.

This lesson wraps the aforementioned protocols and standards together into Application Programming Interfaces, or APIs.

What is an API?

APIs (Application Programming Interfaces) define outward-facing methods for interaction, acting as a bridge between different software components. The goal is to allow communication without knowing anything about each other's internal workings. This abstraction enables you to leverage external services, libraries, and frameworks without building them from scratch.

In the previous lesson introducing web services, you learned that many web services take the form of APIs. Moving forward through this course, the term API will generally be used to describe these - web services that have laid out instructions for how one should communicate by sending and receiving data. APIs will play an integral part of your path forward as a software developer. You will build your own APIs, projects that consume third-party APIs, and everything in between.

There are many types of these APIs, which are usually categorized by the protocols they employ. For example, you might have heard of the Google or Twitter APIs. These two and many others are RESTful APIs. This is the API type you'll learn how to build and interact with.

What is a RESTful API?

You just learned that an API establishes a set of rules for how the server expects instructions, data, and requests formatted - but what specifically does a RESTful API do?

A RESTful API defines any number of **endpoints**. Endpoints are organized by their path in a URL. In the following example, `/user/2034` is an *endpoint* on the API:

https://myapi.com/user/2034

Each endpoint on a server is independent and distinguished from others by its configuration. However, there is one exception - each endpoint path can be defined multiple times using different HTTP methods. Using the example above, there may be a GET endpoint defined at `/user/{userId}` that returns an existing user's information. The same API can provide a POST endpoint also located at `/user` (without specifying a `userId`) which is used to create new users.

Endpoint Configuration

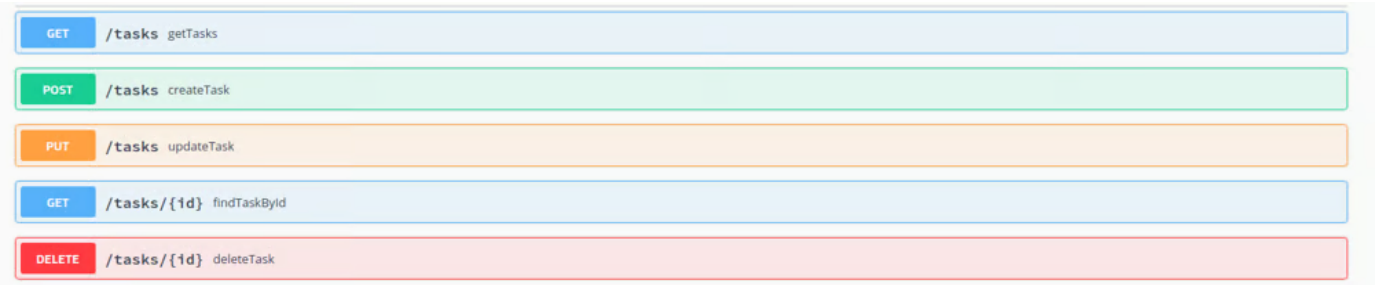
Generally, each endpoint definition has a single specific task and includes the following configuration information:

- Which HTTP method the endpoint will accept.
- What type of action is taken when the endpoint receives a request.
- What information the endpoint needs to process a request.
- What information you can expect in response to a request.

- What type(s) of data the endpoint requires.
- What type(s) of data the endpoint will respond with.
- What methods of authentication (if any) is required.

Each API is different. You'll find that some APIs are intuitive, and some are infuriating. To avoid creating APIs that anger other developers, take the time to think it through and plan it out. But before you can even start to think about creating your own, you need to study some good existing APIs.

The CodingNomads team has set up a demo API that you can interact with. Take a look at the image below.



Each colored box represents an endpoint. You can see that each endpoint specifies what HTTP method and URL path map to it. Hopefully you'll agree that these endpoints are intuitive. Want to get all tasks? Enter `/tasks` as the URL path and use the GET HTTP method. If you want a specific task, add its ID onto the end of the URL etc.

These are just a few endpoints, but modern web services can have hundreds, each with complex business logic being executed each time one is hit. There is no limit, but generally - the more concise, the better the API.

Summary: RESTful APIs

- APIs define outward-facing methods for interaction with different services.
- RESTful APIs define a set of endpoints.
- Endpoints are distinguished from others in the same API by their unique path and the HTTP methods they support.
- An endpoint has a single specific task and includes information on:
    - What HTTP method the endpoint will accept.
    - What type of action is taken when the endpoint receives a request.

- What information the endpoint needs to process a request.
- What information you can expect in response to a request.
- What type(s) of data the endpoint requires.
- What type(s) of data the endpoint will respond with.
- What methods of authentication (if any) is required.

Previous Next → Video: RESTful APIs

Learn more

Beginner - Intermediate Courses

Java Programming

Python Programming

JavaScript Programming

Git & GitHub

SQL + Databases

Intermediate - Advanced Courses

Spring Framework

Data Science + Machine Learning

Deep Learning with Python

Django Web Development

Flask Web Development

Career Tracks

Java Engineering Career Track

Python Web Dev Career Track

Data Science / ML Career Track

Career Services

Resources

About CodingNomads

Corporate Partnerships

Contact us

Blog

Discord