**Request and response objects**

 Estimated Time: 1-2 hours

The role of a web server is to respond to HTTP *requests* from clients with HTTP *responses*. Express provides a [request](http://expressjs.com/en/api.html#req) and a [response](http://expressjs.com/en/api.html#res) object for representing and interacting with HTTP requests and responses. This reading will get you up to speed on the basics of working with HTTP requests and responses in Express apps.

**In this assignment we're going to use**[**Postman**](https://www.getpostman.com/)**to make requests to a live Node server. We'll have occasion to use Postman throughout this course, and it will prove especially useful when we get into topics like cookies and authorization later in the course. Take a moment to download Postman, and keep it open as you read through this assignment.**

HTTP requests ask servers to return a resource available at a particular place (namely, at the host specified in the request headers, plus the path specified in the initial line). The server's job is first to appropriately route the request to the correct request handler (for instance, requests for /current-weather might be handled by a getCurrentWeather function). Second, the request handler needs to know how to translate the information from the request into an appropriate response it can send back.

All requests received by an Express app will have a method, which is used to appropriately route the request. Express supports [a large number of HTTP methods](http://expressjs.com/en/api.html#routing-methods), but the ones you'll most commonly use are GET, POST, PATCH, PUT, and DELETE (which correspond to the Express methods app.get(), app.post(), app.patch(), app.put(), and app.delete(), respectively. The request method along with the request path are used to route the request to the right *request handler*, which is a function that knows how to supply the requested resource.

The response object will contain at minimum an HTTP status code and some headers (for instance, for Content-Length). Unless you explicitly set a different status code, HTTP responses default to 200. If the response contains content, that content goes in the response body.

Let's have a look at some examples. The server.js file in the Glitch server below contains four endpoints, each demonstrating a different key concept related to request and/or response objects. As you read through the remainder of this assignment, follow along by trying out the example requests in Postman.