

Julia Evans

- [About](#)
- [Talks](#)
- [Projects](#)
- [Twitter](#)
- [Github](#)

- [Favorites](#)
- [Zines](#)
- [RSS](#)

curl exercises

Recently I've been interested in how people learn things. I was reading Kathy Sierra's great book [Badass: Making Users Awesome](#). It talks about the idea of *deliberate practice*.

The idea is that you find a small micro-skill that can be learned in maybe 3 sessions of 45 minutes, and focus on learning that micro-skill. So, as an exercise, I was trying to think of a computer skill that I thought could be learned in 3 45-minute sessions.

I thought that making HTTP requests with `curl` might be a skill like that, so here are some curl exercises as an experiment!

what's curl?

`curl` is a command line tool for making HTTP requests. I like it because it's an easy way to test that servers or APIs are doing what I think, but it's a little confusing at first!

Here's a drawing explaining curl's most important command line arguments (which is page 6 of my [Bite Size Networking](#) zine). You can click to make it bigger.

♥ these comics?

buy a collection!

★ wizardzines.com ★

curl

JULIA EVANS
@b0rk

curl



great for testing APIs!

\$ curl wizardzines.com

-H
is for header

good for POST requests:

```
-H "Content-Type: application/json"
```

allow compression:

```
-H "Accept-Encoding: gzip"
```

--data
to POST data!

```
--data '{"name": "julia"}'
```

```
--data @filename.json
```

↑
this way is WAY better
if you have a lot of data
to send!

-i

show response headers

-v, -vv

verbose: show request headers etc

★ copy as cURL ★

Have something in your browser you want to download from the command line?

In Firefox / Chrome:

- Developer Tools
- Network tab
- right click on the thing
- copy as curl

-I

show only response headers (HEAD request)

-k

insecure: don't verify SSL certificates

-X POST

send a POST request instead of a GET (-X PUT etc works too)

--cert, --key

use a client certificate

fluency is valuable

With any command line tool, I think having fluency is really helpful. It's really nice to be able to just type in the thing you need. For example recently I was testing out the Gumroad API and I was able to just type in:

```
curl https://api.gumroad.com/v2/sales \
  -d "access_token=<SECRET>" \
  -X GET -d "before=2016-09-03"
```

and get things working from the command line.

21 curl exercises

These exercises are about understanding how to make different kinds of HTTP requests with curl. They're a little repetitive on purpose. They exercise basically everything I do with curl.

To keep it simple, we're going to make a lot of our requests to the same website: <https://httpbin.org>. httpbin is a service that accepts HTTP requests and then tells you what request you made.

1. Request <https://httpbin.org>
2. Request <https://httpbin.org/anything>. httpbin.org/anything will look at the request you made, parse it, and echo back to you what you requested. curl's default is to make a GET request.
3. Make a POST request to <https://httpbin.org/anything>
4. Make a GET request to <https://httpbin.org/anything>, but this time add some query parameters (set value=panda).
5. Request google's robots.txt file (www.google.com/robots.txt)
6. Make a GET request to <https://httpbin.org/anything> and set the header User-Agent: elephant.

7. Make a DELETE request to <https://httpbin.org/anything>
8. Request <https://httpbin.org/anything> and also get the response headers
9. Make a POST request to <https://httpbin.org/anything> with the JSON body `{"value": "panda"}`
10. Make the same POST request as the previous exercise, but set the Content-Type header to `application/json` (because POST requests need to have a content type that matches their body). Look at the `json` field in the response to see the difference from the previous one.
11. Make a GET request to <https://httpbin.org/anything> and set the header `Accept-Encoding: gzip` (what happens? why?)
12. Put a bunch of a JSON in a file and then make a POST request to <https://httpbin.org/anything> with the JSON in that file as the body
13. Make a request to <https://httpbin.org/image> and set the header `'Accept: image/png'`. Save the output to a PNG file and open the file in an image viewer. Try the same thing with different `Accept:` headers.
14. Make a PUT request to <https://httpbin.org/anything>
15. Request <https://httpbin.org/image/jpeg>, save it to a file, and open that file in your image editor.
16. Request <https://www.twitter.com>. You'll get an empty response. Get curl to show you the response headers too, and try to figure out why the response was empty.
17. Make any request to <https://httpbin.org/anything> and just set some nonsense headers (like `panda: elephant`)
18. Request <https://httpbin.org/status/404> and <https://httpbin.org/status/200>. Request them again and get curl to show the response headers.
19. Request <https://httpbin.org/anything> and set a username and password (with `-u username:password`)
20. Download the Twitter homepage (<https://twitter.com>) in Spanish by setting the `Accept-Language: es-ES` header.
21. Make a request to the Stripe API with curl. (see <https://stripe.com/docs/development> for how, they give you a test API key). Try making exactly the same request to <https://httpbin.org/anything>.

Want a weekly digest of this blog?

Subscribe

[Get your work recognized: write a brag document](#) [git exercises: navigate a repository](#)

[Archives](#)

© Julia Evans. If you like this, you may like [Ulita Ea](#) or, more seriously, this list of [blogs I love](#).

You might also like the [Recurse Center](#), my very favorite programming community ([my posts about it](#))