CS 240: cURL Client for URLs Transcript

[00:00:01] Next thing I'd like to talk about is some tools that you can use when you're developing um web applications using http.

[00:00:09] As we talked about clients send requests to servers and receive responses back from servers.

Start visual description. The professor demonstrates how clients send requests to servers and receive responses back from servers, explaining the basic client-server interaction in web applications. End visual description.

[00:00:16] And as you're building a system that, that works that way.

[00:00:19] Uh And debugging it oftentimes, it's useful to be able to see the http messages flowing back and forth between client and server.

[00:00:28] We've already seen how you can use a web browser and its development tools to see the http requests and responses that are flowing back and forth between a browser and a web server.

[00:00:39] And that's great if, if the client that you're debugging is um runs in a browser, but a lot of http clients actually don't run inside a web browser.

[00:00:51] Uh For example, our chess client will not run in a web browser. And so, it's useful to have tools outside of a web browser that you can do something similar to the DEV tools in a browser.

Start visual description. The professor demonstrates the use of cURL, a command line program, to create an HTTP request and send it to a server, showing the response received from the server. End visual description.

[00:01:02] And so I’m going to talk about a few tools you can use to, to inspect the, the messages flowing back and forth um outside of a web browser the first tool I want to talk about is called CURL, see URL.

[00:01:17] And this is a command line program that you can use to create an HTTP request and send it to a server and get back the response. And um that way you can see what the server is sending you back.

[00:01:30] So CURL stands for client for URL S.

[00:01:34] It’s kind of a strange name but that’s what it stands for people just call it CURL.

[00:01:40] So um CURL is really useful for debugging web, API end points or functions or really anything.

[00:01:48] Um That’s web based uh CURL runs on every operating system.

[00:01:53] And so you can always download and install CURL on whatever machine you’re developing on.

[00:01:57] It’s, it’s universally available and um it just has a lot of advantages because it is universally available.

[00:02:04] It is a command line program. So, it doesn’t have a graphical interface.

[00:02:08] But if, if you don’t mind working with a uh a terminal based program, then uh it’s, it’s very powerful.

[00:02:16] OK.

[00:02:17] So the basic syntax for running CURL is just type in the, the command cURL on the command line and then type in the URL that you want to uh access.

[00:02:28] Now by defer default, CURL does an http get request and so let, let’s just type this command into a command line here. So, I’m going to say CURL byu.edu and it shows us the response that came back from by us web server, which in this case, they’re saying um the URL you tried to access has been moved.

Start visual description. The professor demonstrates typing the cURL command into the command line to access a URL, showing the response from the server, including the status code and the new URL for the resource. End visual description.

[00:02:55] So it’s got a 301-status code which means it’s a, what’s called a redirect. So, it’s been moved.

[00:03:01] And the new URL for this resource is http S colon slash slash wwwwyu.edu.

[00:03:08] So if I really want to get by us home page, I’m going to have to type in cURL http S wwwwiu.edu.

[00:03:28] Well, that didn’t work yet. Let’s try it again.

[00:03:31] OK? It worked that time.

[00:03:34] And so what we get back here is the html code for Buy U’s home page.

[00:03:39] And you can see it’s fairly big.

[00:03:42] Um This is the response body that, that by I use web server sent back.

[00:03:48] You can also see the headers, the status code, the reason phrase, you can see all the different parts and pieces of the http response that comes back.

[00:03:57] But by default, it just shows you the um response body.

[00:04:02] OK? So that gives you a sense of how you can um how a web browser would, you know, make a, get request to a web server and get back a web page.

[00:04:11] But you can also call web API s using http as we said before.

[00:04:15] So let's do another example that's not a web page, but rather it's a web API.

[00:04:20] And so in this case, we're going to call a web API, it's called API dot Chuck Norris dot IO.

Start visual description. The professor demonstrates calling a web API using cURL, specifically the API that returns a Chuck Norris joke related to software development and shows the JSON response received. End visual description.

[00:04:27] And what it does is when you call it, it returns a, a Chuck Norris joke.

[00:04:33] And in this case, we're asking it to return a joke that relates to software development.

[00:04:39] And so if we type that into our terminal, you can see it re responded with a Json object that has a Chuck Norris joke in it says Chuck Norris' log statements are always at the fatal level.

[00:04:59] OK? There's a joke.

[00:05:01] And so that's an example of calling a web API. So, this is not a web page.

[00:05:05] This is just a function you can call on somebody's server.

[00:05:08] I can get back a joke.

[00:05:11] Now by default, like I said, CURL does an http get request.

[00:05:15] If you want to do a different kind of request, you, you can um use the dash X option.

[00:05:23] So if I wanted to do a post, I could do that and then give it a URL. Let's see if BYU works on that.

[00:05:39] It seems like it worked.

[00:05:41] So BYU will return a home page if whether you use a get request or a post request apparently.

[00:05:46] OK? Um So there's lots of command line options that you can use with um cURL.

[00:05:56] So with the dash X option, you can specify which http M method you want to use.

Start visual description. The professor demonstrates various command line options available in cURL, such as specifying the HTTP method, turning on verbose mode, and sending a JSON object in the request body. End visual description.

[00:06:02] Uh dash V would put cURL into a verbose mode. So, if you really want to see all the details of what's happening there, you could use verbose, let, let's try that let's put verbose on the Chuck Norris Web API.

[00:06:22] And you can see here when we turn on verbose mode, we get a lot of details about what's happening.

[00:06:28] It's trying to connect to a particular IP address, it's connected.

[00:06:32] Um It shows you the headers that it's sending to the server.

[00:06:36] Um It shows you uh details about the networking that's happening probably more than you usually want to know.

[00:06:44] And then it also shows you all the headers that came back from the server. It shows you the um status code and then it shows you the response body.

[00:06:54] So if you want more details, you can turn on verbose mode and it shows you more stuff.

[00:07:04] Um You can uh use the dash D option to specify the request body on the command line. So, if I say dash d, um if I want to send a JSON object in the request body, I can just type in that little JSON object right there on the command line.

[00:07:19] Or if I don't want to type my request body on the command line, I can put it in a file and then I can use the data slash dash binary option to tell it to read the request body from a file.

[00:07:34] Um I can put headers on my request using the dash H option.

[00:07:41] Um I can have cURL save the output that comes back from the server in a, in a file.

[00:07:46] If I want to use the dash O option.

[00:07:49] And uh I can dump the headers to a file if I want to use the dash D option.

[00:07:54] So there's, there's tons of options on cURL and you ought to probably read the documentation and see what's available.

[00:07:58] But it has a lot of cool things it can do.

[00:08:05] Another example of a web API call is uh the GitHub web API. So, we've been using GitHub in this class.

[00:08:12] Well, GitHub actually has a web API that you can call to do um operations on your repositories.

Start visual description. The professor demonstrates calling the GitHub web API to get information about a GitHub user, showing the details received in the response. End visual description.

[00:08:19] And so one operation you can do in um with the GitHub API as you can get information about a user.

[00:08:31] So let’s try to get some information about a GitHub user.

[00:08:37] Now, this user’s name is Octo cat.

[00:08:42] So I can call that web API, and this gives me all the details about that user.

[00:08:46] Of course, it doesn’t tell me anything that’s private.

[00:08:49] Um But if I wanted to save that output to a file, I could do dash O give it a file name and it would, it would write that data to a file instead of displaying it in the terminal, then I can go back and look at that file.

[00:09:08] All right.

[00:09:09] So tho those are the basic things you can do with cURL.