mkcohen.com

How The Web Works – In One Easy Lesson

9-11 minutes

Like an infomercial claim (“It slices! It dices!”), this article’s title sounds too good to be true, but it is true – in one article, I’m going to explain how the web works and you will walk away a better informed human being. All you have to do is give me a few minutes of your time. Sound like a deal?

As the video above illustrates, lots of people don’t quite understand some basic things about the web, like what a browser is. A web browser is a program running on your computer (or smart phone, or iPad or…) through which you access the World Wide Web. The browser’s job is to make it possible for you to visit pages on the web. But what’s really happening when you use your browser to access the web?

Let’s start at the beginning…imagine you’re sitting in front of your web browser and you enter a URL (which is a fancy term for the address of a resource on the web). Let’s step through what actually happens when you press the enter key. The first step is that your URL gets parsed by the browser. Parse is a fancy term for dividing something into pieces. If you’re ever at a cocktail party with computer scientists, try to work the word ‘parse’ into

the conversation and everyone will be very impressed with you.

URLs are formatted like this: "<protocol>://<server>/<path>". Let's take a look at a real URL and see how it gets divided into pieces:

http://www.npr.org/series/tiny-desk-concerts/

- the protocol: This is the "how" – it tells your computer which conventions to use when talking to the computer serving the requested page. In this example, the desired protocol is "http", which is a special set of rules for requesting and receiving web content.
- the server: This is the "where" – it tells your computer the name of the computer serving the requested page. In this example, the server is "www.npr.org", which is the name for one or more computers operated by NPR.
- the path: This is the "what" – it indicates which page you're interested in accessing on the requested website. In this example, the path is "*series/tiny-desk-concerts/*", which is the name associated with a particular page among many available at the NPR website.

  For every URL you type, there is a computer just like your computer (although probably a lot more powerful), just waiting to respond to your requests. Actually, most popular web sites are served by large banks of computers called clusters, but to the outside world, such clusters operate like a single, very powerful computer, so it's fine if you want to think about all the pages at npr.org as coming from one giant computer. These computers are called "web servers", because they respond to (i.e. serve)

requests from “clients”, like your browser.

Now that the browser has chopped up your URL into pieces, it can get down to work. The first thing it needs to do is establish a communication session with the requested server. But first it needs to figure out how to reach that server on the internet. I’m going to let you in on a little secret: inside the internet, when computers talk to each other, they don’t use the nice, human-friendly names we’re used to, like ‘espn.com’. They use boring-looking sequences of numbers, like 192.168.144.227. These numbers are called IP (Internet Protocol) addresses. Every computer on the internet, including the computer you’re using right now to view this site, is assigned a unique IP address. Would you like to know what your IP address is? Click on this link: http://whatismyipaddress.com/ and you can see your very own personal IP address, as well as some other information about your computer. You’ve been using your computer for how long? And you’re only now learning it’s real name!

Let’s say you want to call your mother. What do you do if you don’t know her phone number? (and, by the way, shame on you for that!) You look it up in the phone book. Or at least that’s what we did in the stone age when we had phone books – now you

might look it up online. The phone book is a great analogy for what goes on when your browser wants to connect to a server it knows only by name – it needs to find the IP address associated with that name. The way it does that is by consulting a special resource called the DNS (Domain Name System). DNS is the internet's "phone book", so to speak. It's how clients, like your web browser, convert a server name into its corresponding IP address.

Want to look up a name in DNS yourself? Visit http://www.webmaster-toolkit.com/dns-query.shtml and enter any server name you like. I just entered "www.npr.org" and found out NPR's IP address is 216.35.221.76. Here's another cool thing… your browser can use addresses just as well as names. Open a new browser window and enter npr.org's address (or just click on this link: http://216.35.221.76). Your browser sees that and says "Wow, this user gave me an IP address so I can skip the hassle of looking up a name in DNS and just connect directly to the address provided".

What happens next? Your computer makes a connection to the server's IP address and the server accepts the connection, sort of like the way you call your Mom and she answers the phone. After the connection is established, your computer sends something called an HTTP request (more on this later) and the server does one of two things: if it can find the page you requested, it returns it in an HTTP response. If the server can't find the page you requested, it returns a special "404 page not found" response, which we all see from time to time when we mistype a URL.

On the world wide web, computers don't communicate with words, they use protocols like HTTP (HyperText Transfer Protocol). HTTP is a way to structure requests for web resources (and the corresponding responses) so that that they can be understood clearly and unambiguously by a computer. The request/response between your browser and the server is similar to this scenario: after your Mom answers the phone, you say "hey, Mom, can you give me your recipe for that delicious Fritos casserole?". That's very similar to an HTTP request for a particular object (the casserole recipe). In response, your Mom does one of two things. She might say "oh, sure, I have it right here – first you pre-heat the over to 425 degrees…", which is like the HTTP response above. Or, she might say, "I'm sorry, but I can't find that recipe, I must have misplaced it", which is the human equivalent of a "404 page not found" HTTP response.

In addition to using a protocol to manage the transfer for information, the actual content that gets transferred and presented by your browser also follows a very precise format called HTML (HyperText Markup Language). Here's an example of a very simple HTML document:

```
<h1>Here's a picture of my dog:</h1>
<img src="wp-content/uploads/2011/02/meiko.jpg" />
<p>His name is Meiko. As you can see, he is quite awesome.
```

Just to give a taste of what you can do with HTML, the <h1> and </h1> “tags”, as they are called, bracket some text to be printed as a heading, and the <img> tag identifies an image or a picture to be displayed. The <p> tag starts a new paragraph.

I’ve created a file containing the HTML document above at the path “/dog.html” on my server (mkcohen.com). I’ve set it up to use HTTP as the transport protocol, so putting those three pieces together, the entire URL for accessing my document above would be: http://mkcohen.com/dog.html. Go ahead, click on that link and see what happens (I’ve programmed it to open a new browser window so you won’t lose your place in this article). Here’s a review of what just happened:

1. You told your browser you wanted to visit a particular URL (http://mkcohen.com/dog.html).
2. Your browser parsed the URL into three pieces: the protocol (HTTP), the server (mkcohen.com) and the path (dog.html).
3. Your browser used the DNS system to convert the server’s user-friendly name (mkcohen.com) into my server’s internet protocol address (174.121.79.148).
4. Your browser made a connection to my server’s IP address.
5. Your browser sent my server an HTTP request asking for a copy of the HTML document stored at dog.html.
6. My server found the requested HTML document and returned it to your browser via an HTTP response.
7. Your browser received the response.
8. Your browser interpreted and displayed the HTML document

contained in the response. At this point, you struggled to contain your joy as the magnificently handsome Meiko appeared on your screen.

9. Your browser dropped the connection to my server, terminating the session.

   Of course, there's a lot more to this story but these are the basic, fundamental things that happen every time you click on a link or visit a web page. You may not be ready to build your own browser but I hope you now have a better understanding of how the web works. Leave me a note below if you have any questions or comments.

   P.S. No dogs were harmed in the making of this article.