How the Web Works

Blase Ur and Grant Ho and David Cash

April 28th, 2026

CMSC 23200

THE UNIVERSITY OF CHICAGO

The Anatomy of a URL

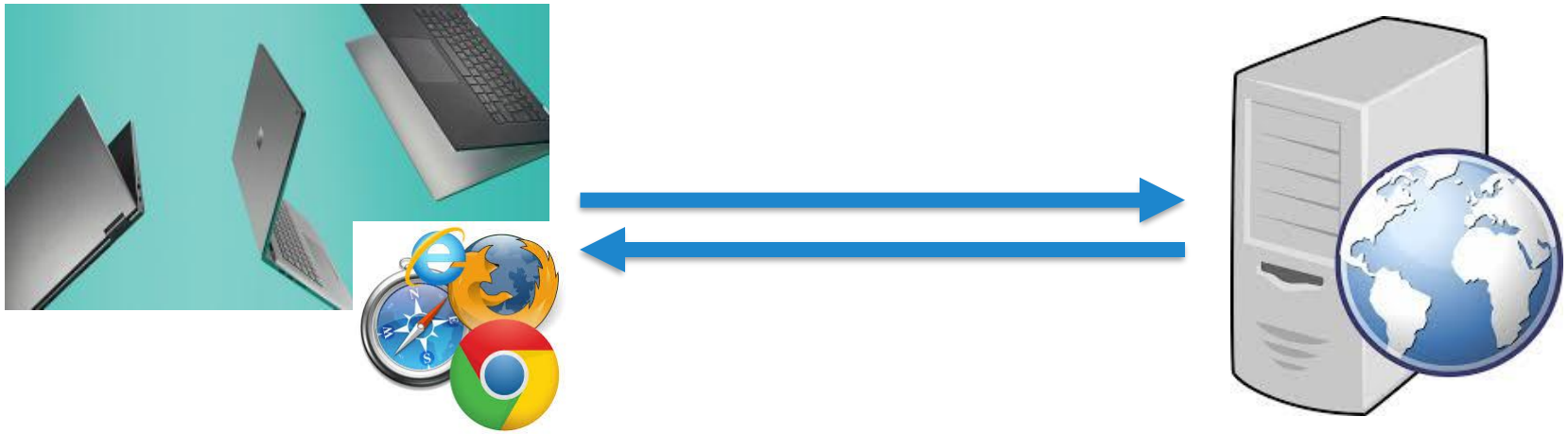
- https://www.uchicago.edu/fun/funthings.htm

The Anatomy of a URL

- https://www.uchicago.edu/fun/funthings.htm
  - Protocol: https
  - Hostname: www.uchicago.edu
    - .edu is the top level domain (TLD)
  - Path: /fun/funthings.html

Your interface to the web

- Your web browser contacts a web server



Typing Something into a Browser:

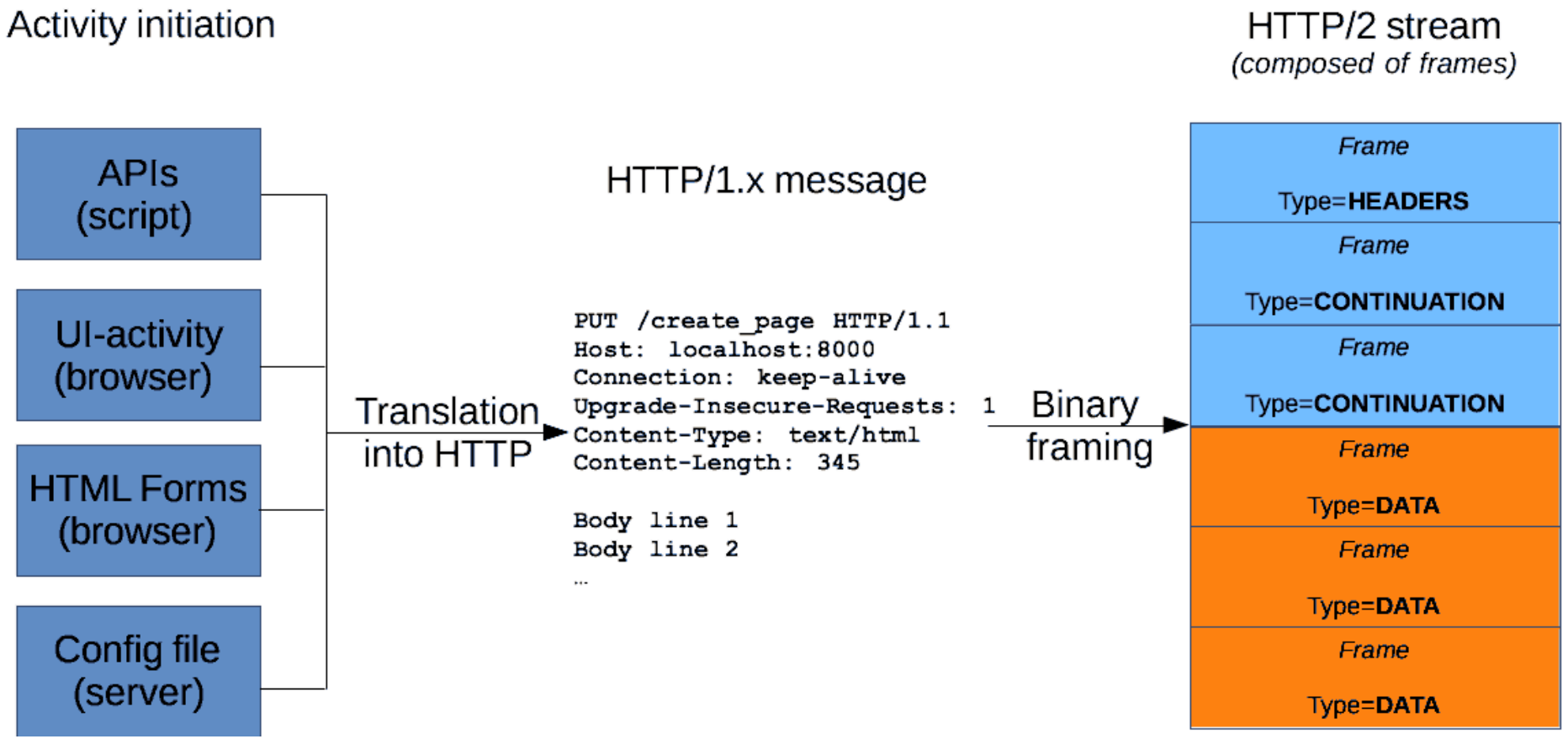
- DNS (domain name service)
  - www.cs.uchicago.edu resolves to IP address 128.135.164.125
- https://www.cs.uchicago.edu/
  - Protocol: https
  - Hostname: www.cs.uchicago.edu
  - Default file name (since none is listed): index.html (and similar)

HTTP Request

- HTTP = Hypertext Transfer Protocol
- Start line: method, target, protocol version
  - GET /index.html HTTP/1.1
  - Method: GET, PUT, POST, HEAD, OPTIONS
- HTTP Headers
  - Host, User-agent, Referer, many others
  - https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers
- Body (not needed for GET, etc.)
- In Firefox: F12, “Network” to see HTTP requests

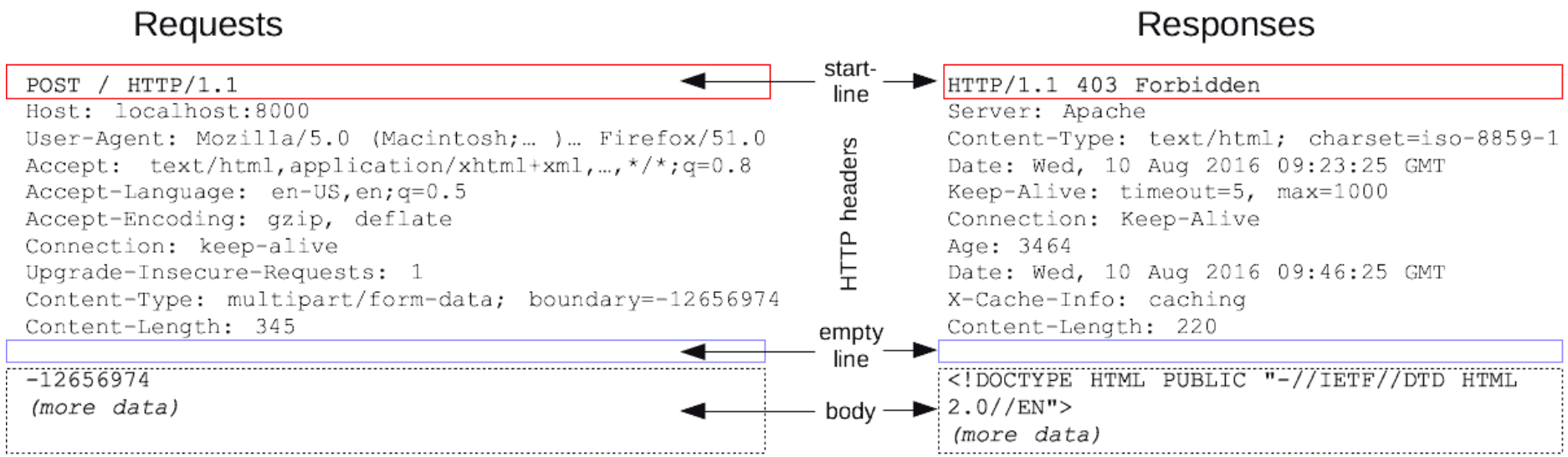
HTTP Request

- GET /index.html HTTP/1.1



From https://developer.mozilla.org/en-US/docs/Web/HTTP/Messages

HTTP



From https://developer.mozilla.org/en-US/docs/Web/HTTP/Messages

HTTP Response

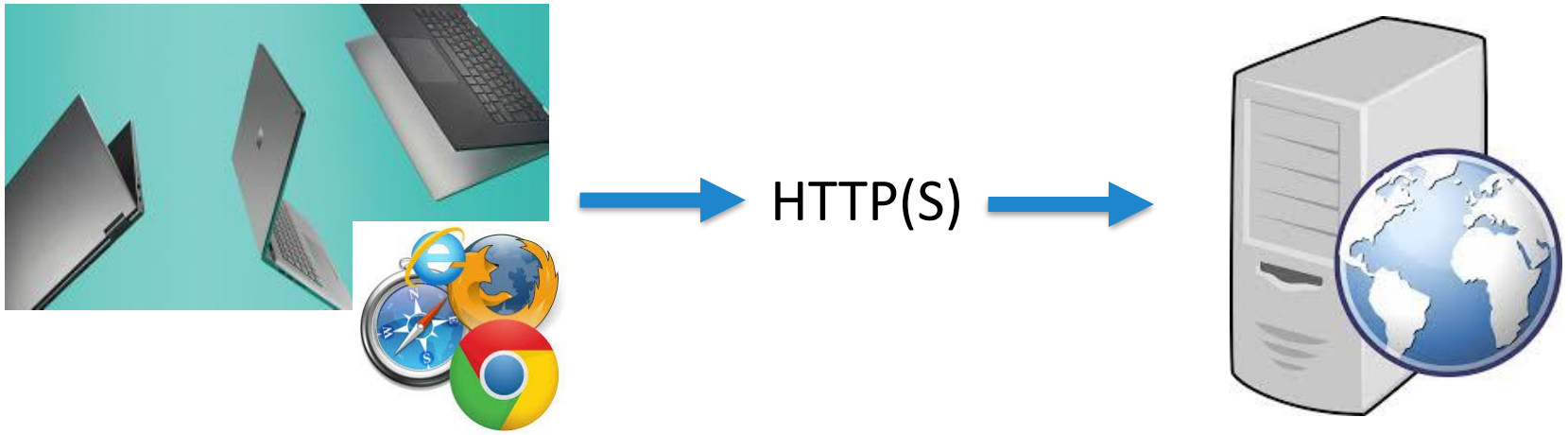
- Status: https://developer.mozilla.org/en-US/docs/Web/HTTP/Status
  - 200 (OK)
  - 404 (not found)
  - 301 (moved permanently)
  - 302 (moved temporarily)
- HTTP Headers
- Body

HTTPS

- Simply an HTTP request sent over TLS!
  - That is, the request and response are encrypted
- An extension of HTTP over TLS (i.e., the request/response itself is encrypted)
- Which CAs (certificate authorities) does your browser trust?
  - Firefox: Options → Privacy & Security → (all the way at the bottom) View Certificates

A 10,000 Foot View of Technologies

- Where things run:



HTML / CSS

JavaScript
(Angular/React)

Browser Extensions

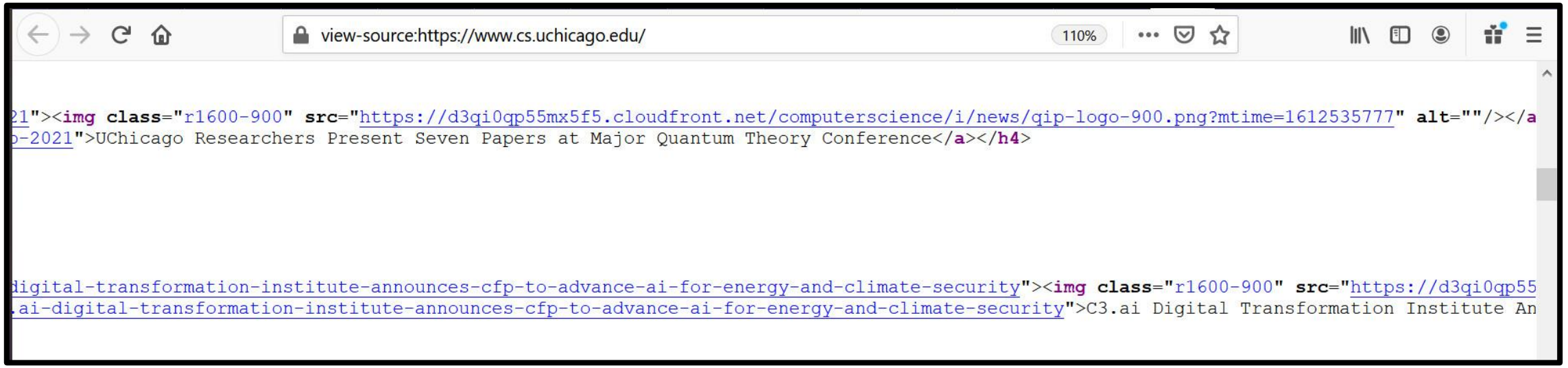
Django (Python) / CGI (Perl) /
PHP / Node.js / Ruby on Rails

Databases (MySQL)

The Anatomy of a Webpage

- view-source:https://www.cs.uchicago.edu/
- HTML (hypertext markup language)
  - Formatting of a page
  - All sorts of formatting: Hi
  - Links: Click here
  - Pictures:
  - Forms
  - Audio/video

The Anatomy of a Webpage



The Anatomy of a Webpage

- CSS (cascading style sheets)

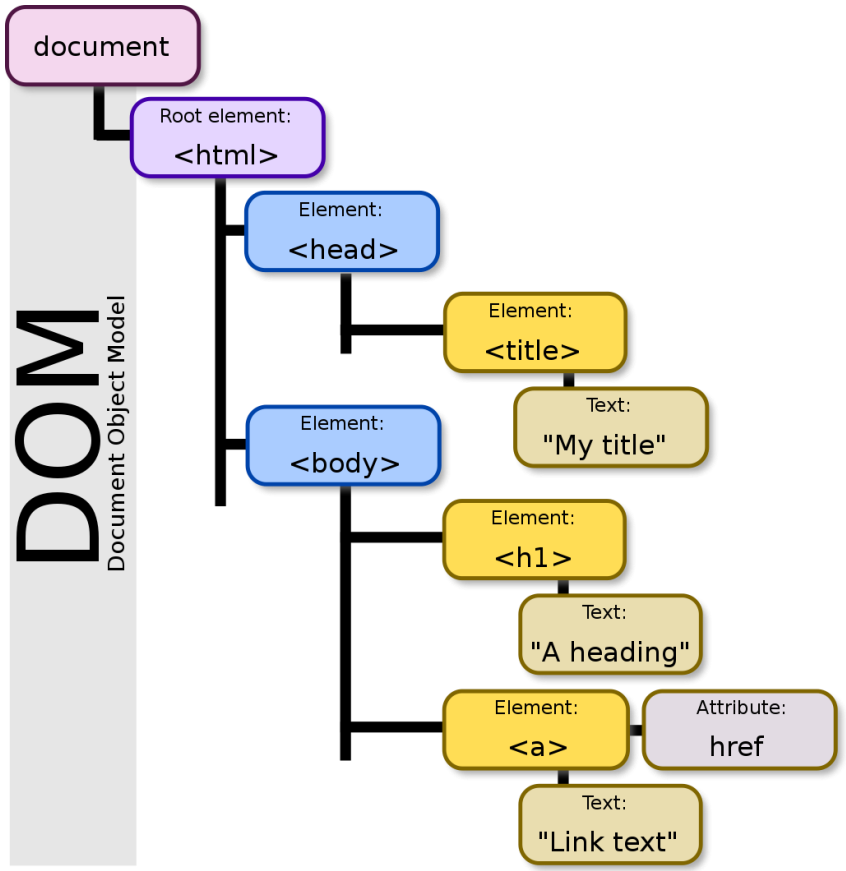
  <link href="/css/main.css?updated=20181020002547" rel="stylesheet" media="all">

  view-source:https://www.cs.uchicago.edu/css/main.css?updated=20181020002547

- #id (*intended* to be unique)
- .class (not intended to be unique)

The Anatomy of a Webpage

- DOM (document object model)



JavaScript

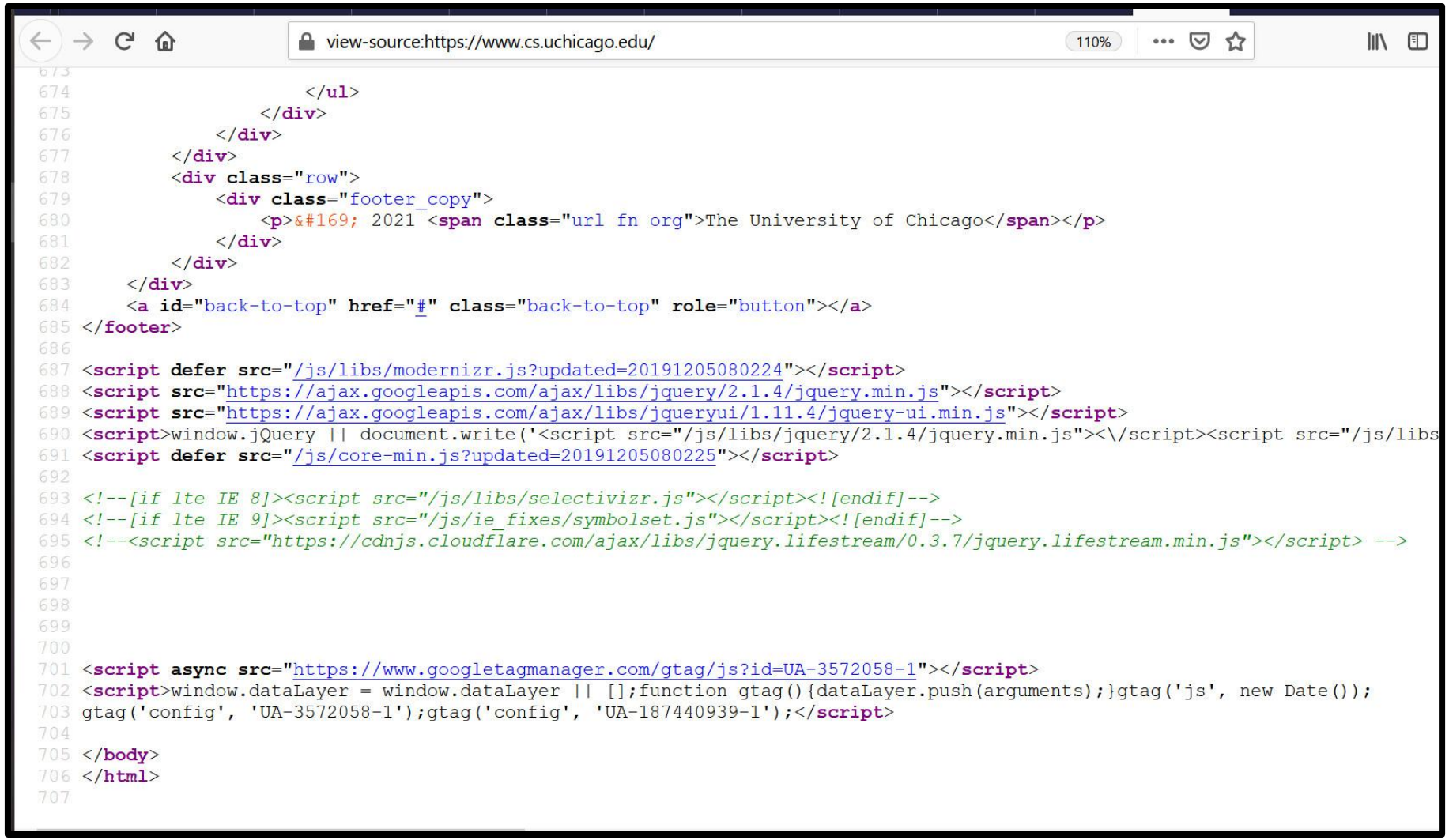
Interactive Pages?

- JavaScript!
  - The core idea: Let's run code on the **client's** computer
- Math, variables, control structures
- Imperative, object-oriented, or functional
- Modify the DOM
- Request data (e.g., through Fetch)
- Can be multi-threaded (web workers)

Common Javascript Libraries

- JQuery (old!)
  - $(".test").hide() hides all elements with class="test"
- JQueryUI
- Bootstrap
- Angular / React (more modern)
- Google Analytics

Importing Javascript Libraries




```
                    </ul>
                </div>
            </div>
        </div>
        <div class="row">
            <div class="footer_copy">
                <p>&#169; 2021 <span class="url fn org">The University of Chicago</span></p>
            </div>
        </div>
    </div>
    <a id="back-to-top" href="#" class="back-to-top" role="button"></a>
</footer>

<script defer src="/js/libs/modernizr.js?updated=20191205080224"></script>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.4/jquery.min.js"></script>
<script src="https://ajax.googleapis.com/ajax/libs/jqueryui/1.11.4/jquery-ui.min.js"></script>
<script>window.jQuery || document.write('<script src="/js/libs/jquery/2.1.4/jquery.min.js"><\/script><script src="/js/libs
<script defer src="/js/core-min.js?updated=20191205080225"></script>

<!--[if lte IE 8]><script src="/js/libs/selectivizr.js"></script><![endif]-->
<!--[if lte IE 9]><script src="/js/ie_fixes/symbolset.js"></script><![endif]-->
<!--<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery.lifestream/0.3.7/jquery.lifestream.min.js"></script> -->



<script async src="https://www.googletagmanager.com/gtag/js?id=UA-3572058-1"></script>
<script>window.dataLayer = window.dataLayer || [];function gtag(){dataLayer.push(arguments);}gtag('js', new Date());
gtag('config', 'UA-3572058-1');gtag('config', 'UA-187440939-1');</script>

</body>
</html>
```

Do You Have the Right .js File?

- Subresource integrity (SRI): https://developer.mozilla.org/en-US/docs/Web/Security/Subresource_Integrity
- <script src="https://example.com/example-framework.js" integrity="sha384-oqVuAfXRKap7fdgcCY5uykM6+R9GqQ8K/uxy9rx7HNQlGYl1kPzQho1wx4JwY8wC"></script>
- cat FILENAME.js | openssl dgst -sha384 -binary | openssl base64 –A

Patching JavaScript Libraries

- Many outdated (and sometimes vulnerable) JavaScript libraries continue to be used
- Very complex chain of dependencies!
  - How do you determine if a given change is for good or evil?

Core Web Defense: Same-Origin Policy

Same-Origin Policy

- Prevent malicious DOM access
- Origin = URI scheme, host name, port
- Only if origin that loaded script matches can a script access the DOM
  - It's not where the script ultimately comes from, but which origin *loads* the script that matters!!!

Same-Origin Policy (SOP)

https://developer.mozilla.org/en-US/docs/Web/Security/Same-origin_policy

Definition of an origin

Two URLs have the *same origin* if the protocol, port (if specified), and host are the same for both. You may see this referenced as the "scheme/host/port tuple", or just "tuple". (A "tuple" is a set of items that together comprise a whole — a generic form for double/triple/quadruple /quintuple/etc.)

The following table gives examples of origin comparisons with the URL `http://store.company.com/dir/page.html`:

URL	Outcome	Reason
`http://store.company.com/dir2/other.html`	Same origin	Only the path differs
`http://store.company.com/dir/inner /another.html`	Same origin	Only the path differs
`https://store.company.com/page.html`	Failure	Different protocol
`http://store.company.com:81/dir/page.html`	Failure	Different port (`http://` is port 80 by default)
`http://news.company.com/dir/page.html`	Failure	Different host

Iframes (Inline Frames)

- Enable you to embed a webpage inside another webpage

Image from https://www.thoughtco.com/when-to-use-iframes-3468667