The Web

Overview

The World Wide Web was invented by Tim Berners-Lee in 1989 as a way to provide a global information system on the Internet. Originally, pages consisted of basic HTML formatting, with links, as shown below, in the first web page ever published.

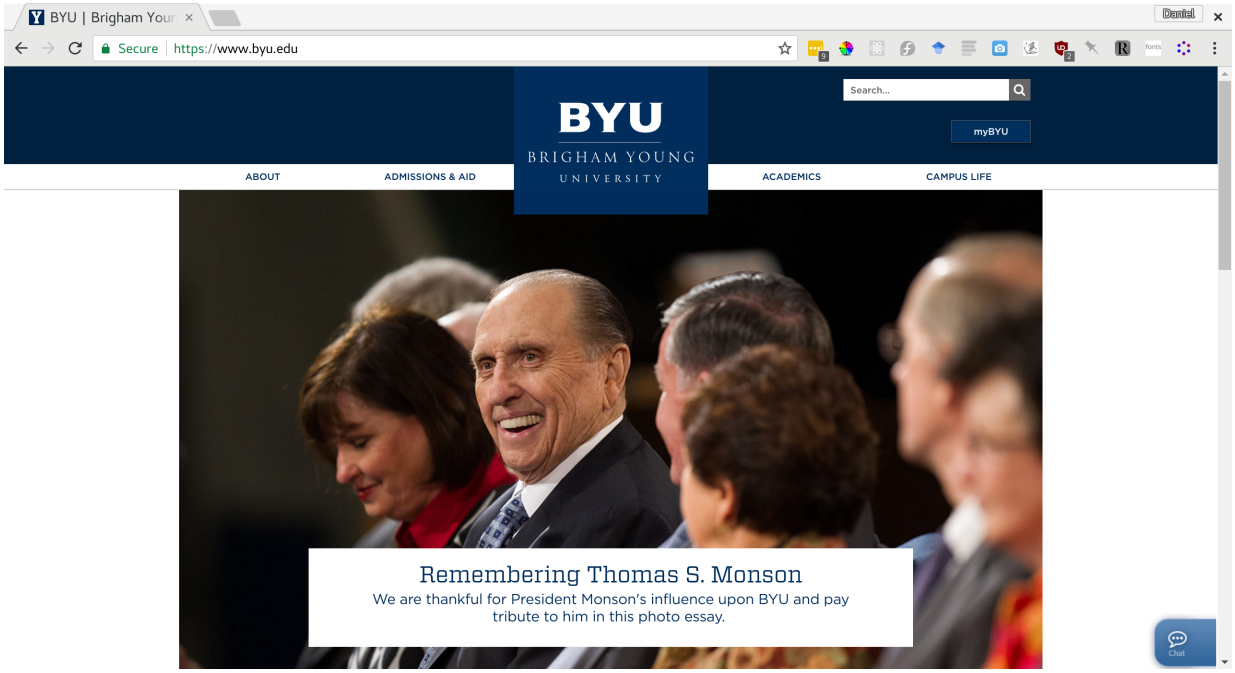
World Wide Web

The WorldWideWeb (W3) is a wide-area hypermedia information retrieval initiative aiming to give universal access to a large universe of documents.

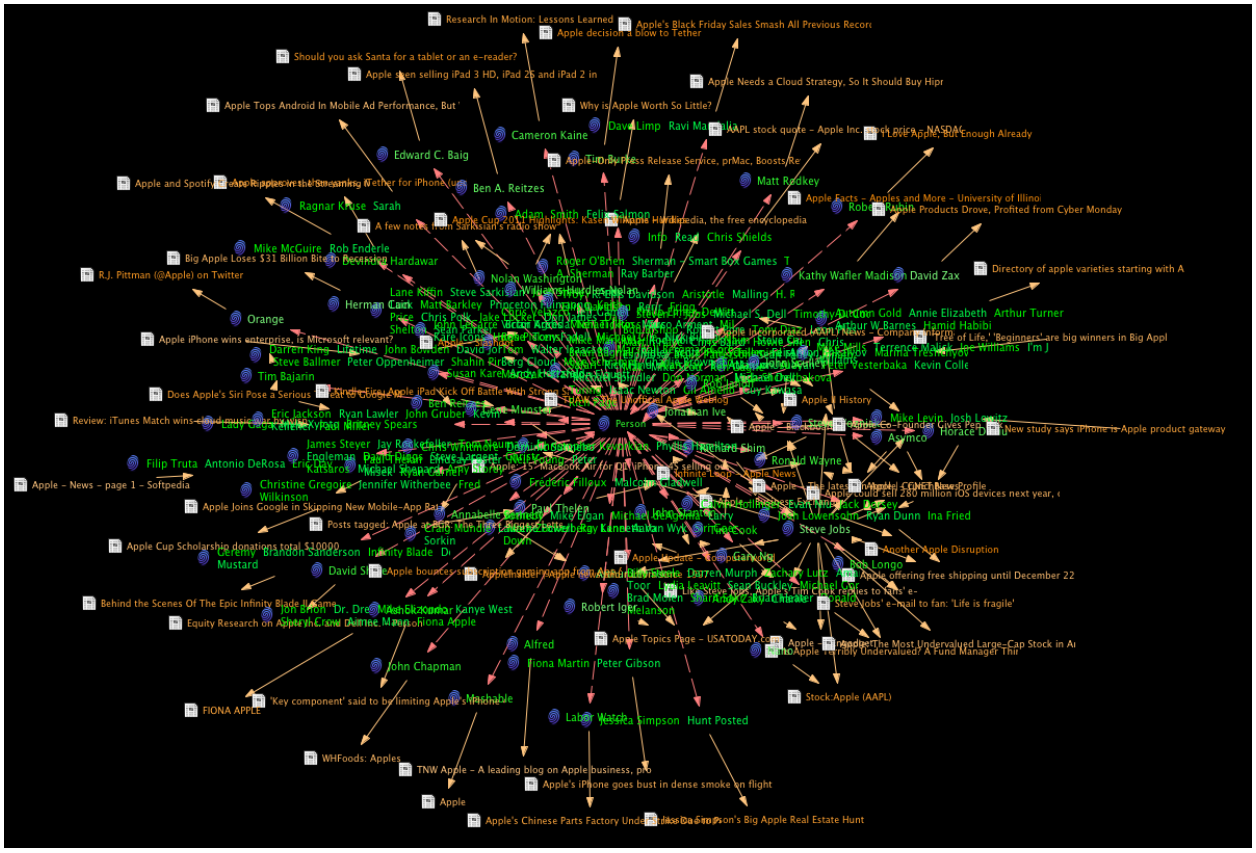
Everything there is online about W3 is linked directly or indirectly to this document, including an executive summary of the project, Mailing lists , Policy , November's W3 news , Frequently Asked Questions .

What's out there?
Pointers to the world's online information, subjects , W3 servers, etc.
Help
on the browser you are using
Software Products
A list of W3 project components and their current state. (e.g. Line Mode ,X11 Viola , NeXTStep , Servers , Tools , Mail robot , Library)
Technical
Details of protocols, formats, program internals etc
Bibliography
Paper documentation on W3 and references.
People
A list of some people involved in the project.
History
A summary of the history of the project.
How can I help ?
If you would like to support the web..
Getting code
Getting the code by anonymous FTP , etc.

Today, pages are much more complex and rich with graphics. For example, see the BYU home page below:



Since its invention, the web has grown at an exponential pace, comprising 4.54 billion pages as of 2018. As can be imagined, visualizing this complex web is extremely challenging. Below is a very simple visualization of one search of pages for "Apple" on Google:



You can view a more ambitious attempt to visualize the web at [http://internet-map.net/ (http://internet-map.net/)](http://internet-map.net/) .

HTML and CSS

Today, content of a web page is formatted primarily using HTML and CSS. HTML controls the structure of the page, for example whether some text is a header, a paragraph, or a bulleted list. CSS controls the format of the page, for example whether some content is *italic,* **bold**, has a blue background with white text, or a colored box.

For example, in HTML you would use to mark the start of a paragraph and to mark the end of a paragraph:

```
<p>A paragraph</p>
```

You could then use CSS to control the style of the paragraph:

```
p {
  font-weight: bold;
}
```

Originally, some formatting was included in HTML, including tags for bold and italic, and these are still used today. However, most formatting has moved to CSS. In addition, as the web has grown, so has the desire for more creative styling of web pages. Today, you can create fairly complex layouts using HTML and CSS.

As an example, view the BYU home page and then type "control-U" (most browsers) to view the HTML source of a page. Likewise, you can right-click in most browsers and select "Inspect" (Chrome) or "Inspect Element" (Firefox) to view the CSS rules that are controlling the display of elements on a page.

For more creative examples, see designs in the **Zen Garden (http://www.mezzoblue.com/zengarden/alldesigns/)** .

JavaScript

JavaScript adds the ability to run code with a web page. For example, JavaScript could be used to:

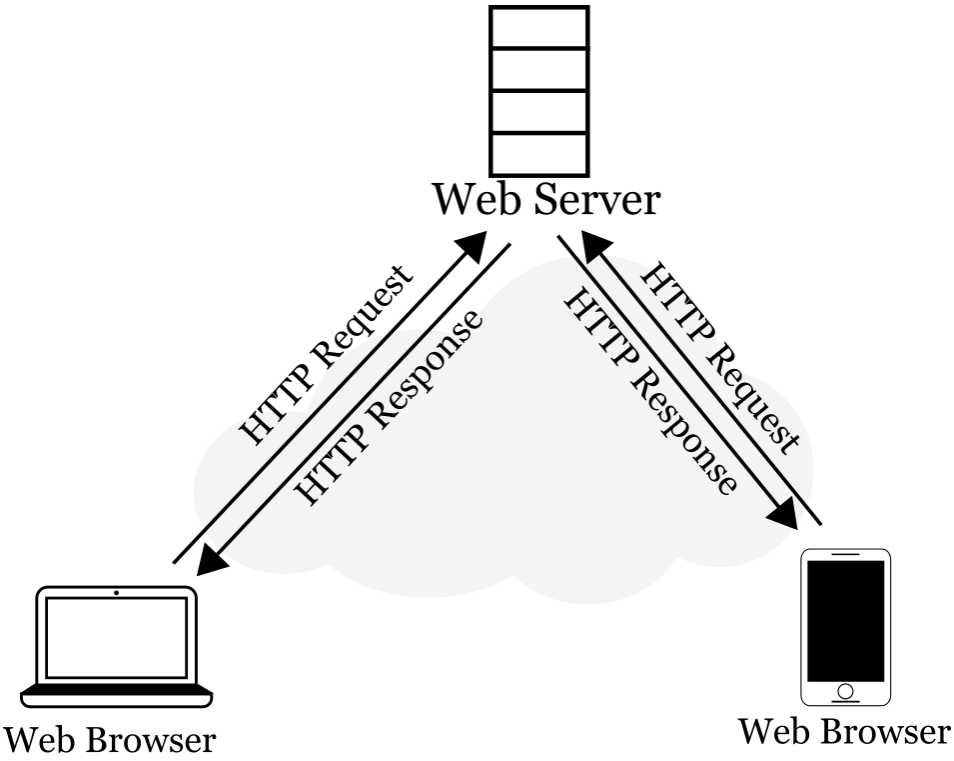
- submit a username and password to a server to login a user to the website,
- show additional information about an book for sale when clicking a link labeled "details",
- zoom in or scroll around on an image when the user clicks on the image or moves their mouse around,
- load additional parts of a map as the user clicks around,

and many other things.

HTTP

Web browsers use HTTP to request objects from web servers. For each object it wants to download, the browser sends an HTTP request and the server sends an HTTP response.

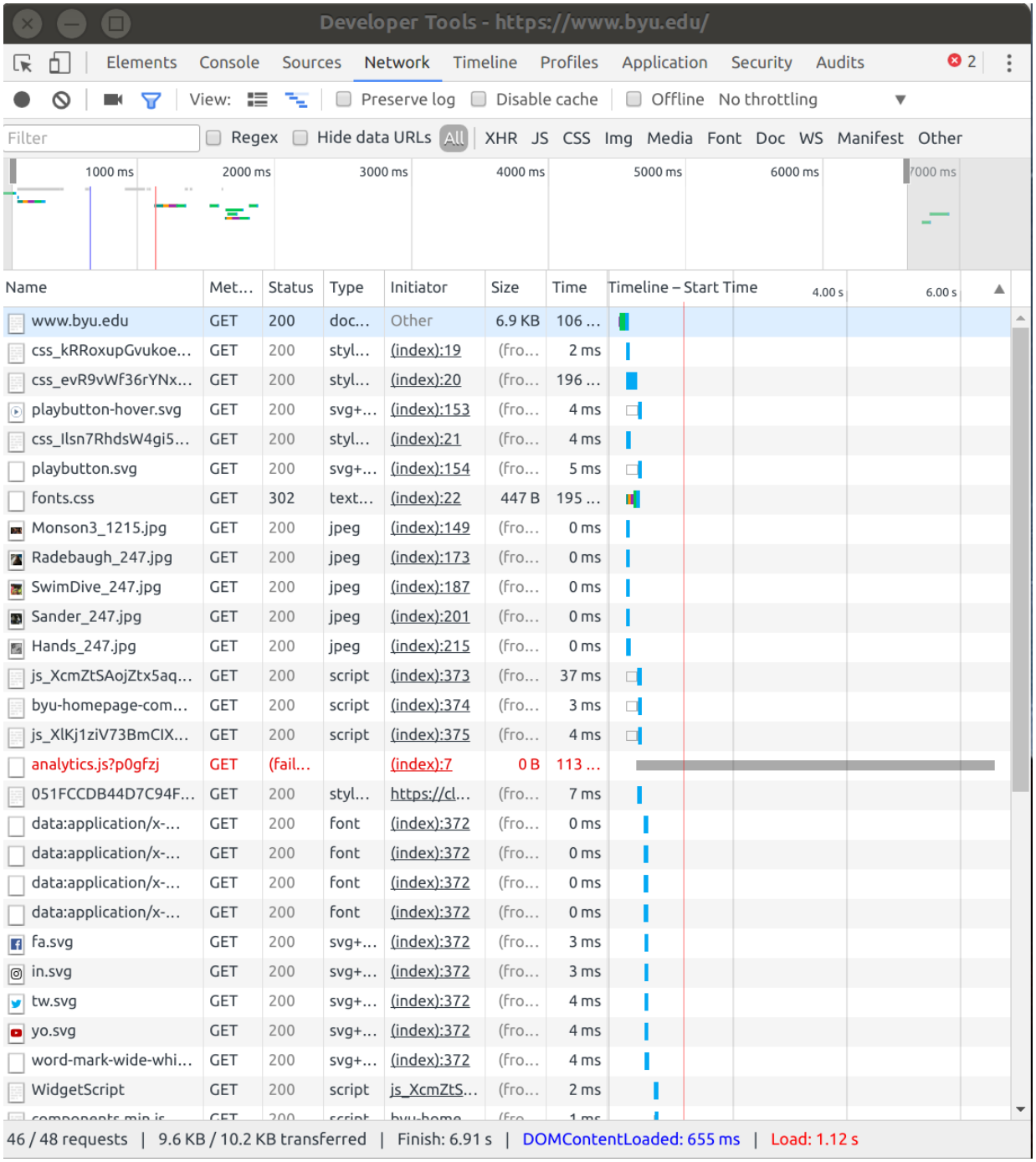
(https://byu.instructure.com/courses/5724/files/1199351/download?wrap=1)



Fetching a single web page requires first requesting the container object (usually an HTML page called index.html) and then requesting additional objects that the container links to and that are necessary to display the page. This typically includes additional HTML, CSS, images, JavaScript.

For example, when fetching the BYU home page, requires fetching four font files, five JPG images, three PNG images, 10 SVG images, four CSS files, thirteen JavaScript files, and three JSON files.

You can see a trace of the HTTP requests and responses made for a web page by opening the developer tools in your browser and using the network tab. For example, you can open this in Chrome under "More Tools" or use control-shift-I. Below is a screenshot of what my browser loaded. You can see one request for a tracking script, shown in red, that was blocked by my ad blocker.



You can click on individual requests and view details of the HTTP request and response. A simple HTTP request looks like this:

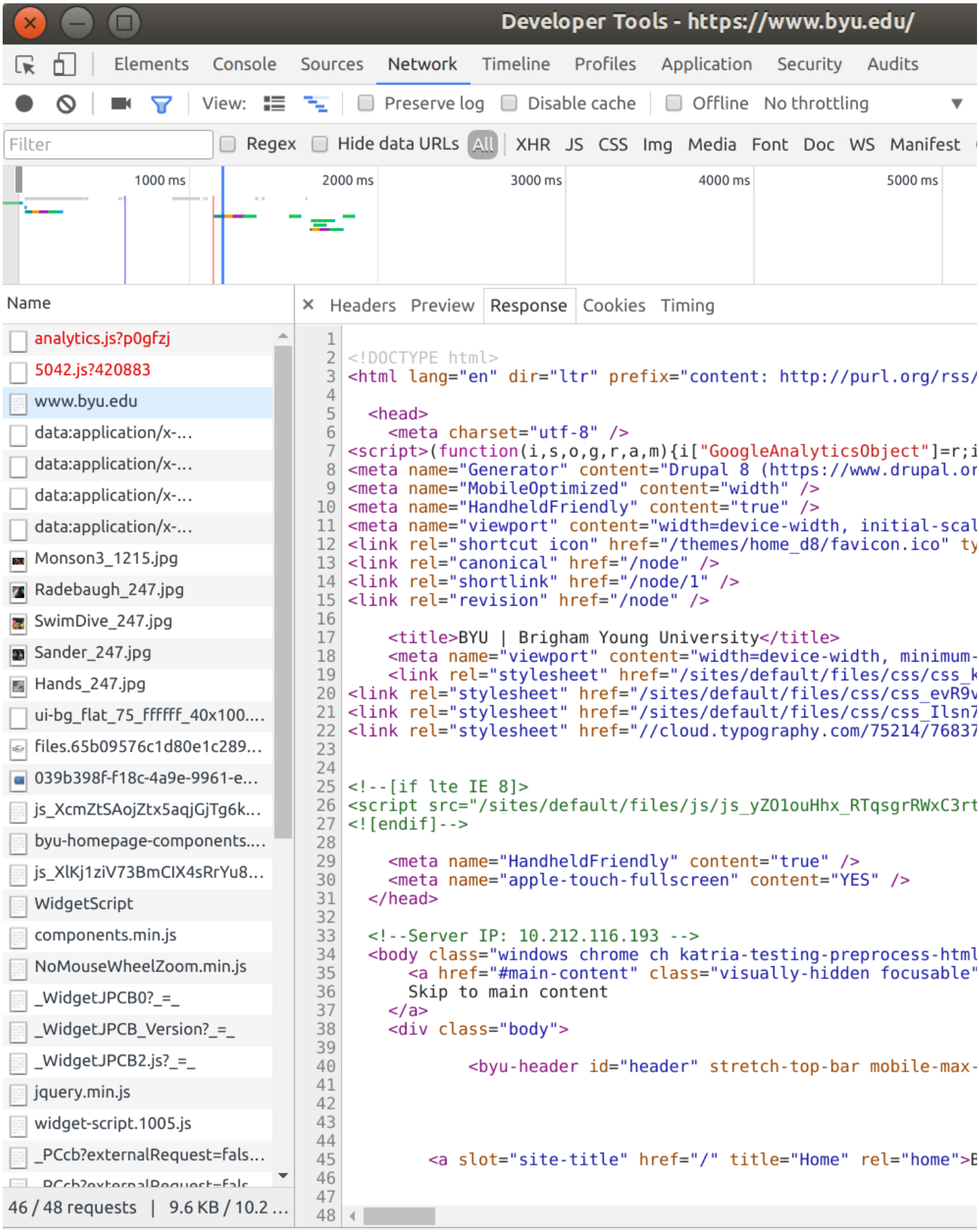
```
GET / HTTP/1.1
Host: www.byu.edu
Accept: text/html,application/xhtml+xml,application/xml
Accept-Encoding: gzip,deflate,sdch
Accept-Language: en-US,en;q=0.8
Connection:keep-alive
User-Agent:Mozilla/5.0 (X11; Linux x86_64) AppleWebKit
/537.36 (KHTML, like Gecko) Chrome/29.0.1547.76 Safari/537.36
```

The first two lines are the most important part. This is the browser requesting a particular file ("/", which is the root of the web server's files, usually index.html), using HTTP version 1.1, from the host **www.byu.edu (http://www.byu.edu)**. Other headers indicate what formats and languages the browser can accept, the browser asks the server to keep the TCP connection alive (so it can send more requests), and the browser tells the server what kind of web browser it is using.

A simple HTTP response looks like this:

```
HTTP/1.1 200 OK
Connection:Keep-Alive
Date:Fri, 05 Jan 2018 19:21:29 GMT
ETag:"1515172376-gzip"
Keep-Alive:timeout=2
Server:Apache/2.4.27 (Amazon) PHP/7.1.11
```

The most important line is the first one, which indicates the request succeeded, using the "200 OK" code. The server also tells the browser it will keep the TCP connection alive for at most 2 seconds in between requests, provides the current date, provides a unique tag for the content, and tells it what version of web server it is running. After this message comes the actual file the web server is sending the browser, which is the HTML formatted container object for the web page. Below is a screenshot of what that web page looks like:



When a web server sends an HTTP response, it includes a code indicating the status of the response. Some of the codes you may encounter during the class include:

- **200 OK**: the request was successful and the response includes the object requested.
- **304 Not Modified**: the request asked if the object has been modified, and it has not been modified; this happens when you hit refresh in the browser.
- **400 Bad Request**: the request was not formatted properly.
- **403 Forbidden**: you do not have permission to access this object. This may happen if the web server is not configured correctly, or if you need to login to access the object, or if you are logged in but trying to access something that you don't have the right to access (e.g. someone else's account information).
- **404 Not Found**: the web server does not have the requested object or file.
- **500 Internal Server Error**: the web server had a failure when trying to retrieve the requested object. This happens often when you are writing a web application and you have a bug in your code.

Learn More

If you want to learn more about HTML, CSS, JavaScript, and how to build web applications, you are in the right class! We will cover a lot more of this in greater detail through the semester.

If you want to learn more about HTTP, we will cover some of this in class but CS 324 and CS 460 cover this material in much more depth. Mozilla has some good resources at **https://developer.mozilla.org/en-US/docs/Web/HTTP (https://developer.mozilla.org/en-US/docs/Web/HTTP)** .