Jerry Cain and Eric Roberts
CS 106J

Handout #47
May 31, 2017

JavaScript and the Web

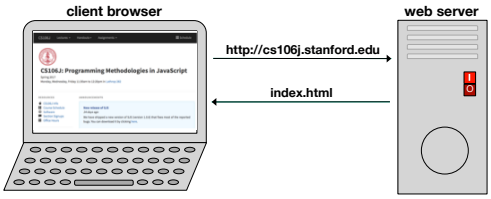
JavaScript and the Web

Jerry Cain and Eric Roberts
CS 106J
May 31, 2017

The History of the World Wide Web

- The ideas behind the web are much older than the web itself.
  - In the early 20th century, the Belgian bibliographer Paul Otlet envisioned a universal catalogue that would provide access to all the world's information in an interconnected structure.
  - In 1945, the director of the wartime science program Vannevar Bush published an article entitled "As We May Think," which envisioned an electronic archive of linked information.
  - In the early 1960s, computer visionary Ted Nelson coined the terms *hyperlink* and *hypermedia*.
- The modern web was developed in 1989 by Tim Berners-Lee at CERN, the European particle physics laboratory in Geneva, Switzerland. Berners-Lee developed the first version of the ***Hypertext Markup Language*** (***HTML***).
- Use of the web grew quickly after the release of Mosaic browser in 1993 and Netscape Navigator in 1994.

The Client-Server Model



1. The user starts a web browser.
2. The user requests a web page.
3. The browser sends a network request for the page.
4. The server sends back a text file containing the HTML.
5. The browser interprets the HTML and renders the page image.

The Three Central Web Technologies

- Modern web pages depend on three technological tools: ***HTML*** (Hypertext Markup Language), ***CSS*** (Cascading Style Sheets), and JavaScript.
- These tools are used to control different aspects of the page:
  - HTML is used to specify content and structure.
  - CSS is used to control appearance and formatting.
  - JavaScript is used to animate the page.
- For this week in CS 106J, our goal is to teach you just enough about HTML—and an even smaller amount of CSS—to build simple web pages.

The Structure of an HTML File

- An HTML file consists of the text to be displayed on the page, interspersed with various commands enclosed in angle brackets, which are known as ***tags***.
- HTML tags usually occur in pairs. The opening tag begins with the name of the tag. The corresponding closing tag has the same name preceded by a slash. The effect of the tag applies to everything between the opening and closing tag.
- As an example, text enclosed between the tags **`<b>`** and **`</b>`** appears in **boldface**. Similarly text enclosed between the tags **`<i>`** and **`</i>`** appears in *italics*.
- A few HTML tags have no closing form, such as the **`<br />`** tag that signals a line break. By convention, such tags are written with a slash before the closing bracket to emphasize that the tag is automatically closed.

Standard `index.html` Pattern

- The main HTML file for a web page is called **`index.html`**. Much of the format of the **`index.html`** file is standardized:
  - Every file begins with a **`<!DOCTYPE html>`** tag
  - The entire content is enclosed in **`<html>`** and **`</html>`** tags.
  - The file begins with a **`<head>`** section that specifies the title.
  - The file includes a **`<body>`** section that specifies the contents.

```
<!DOCTYPE html>
<html>
<head>
<title>title of the page</title>
</head>
<body>
Contents of the page.
</body>
</html>
```

A Sample HTML Page

```
<!DOCTYPE html>
<html>
<head>
<title>HTML - Wikipedia</title>
</head>
<body>
<h1>HTML</h1>
<hr />
<p><b>Hypertext Markup Language</b> (<b>HTML</b>) is the
standard markup language for creating web pages and web
applications.
With Cascading Style Sheets (CSS) and JavaScript it
forms a triad of cornerstone technologies for the
<a href="http://en.wikipedia.org/wiki/World_Wide_Web">
World Wide Web</a>.
Web browsers receive HTML documents from a webserver or
from local storage and render them into multimedia web
pages.</p>
</body>
</html>
```

The Page Displayed by the Browser

Tags that Define Document Sections

<!DOCTYPE>	Defines the document type.
<html>	Encloses the HTML code.
<head>	Encloses the header section.
<title>	Defines the document title.
<body>	Encloses the body of the HTML document.
<p>	Defines a text paragraph.
<h1> . . . **<h6>**	Defines a header at level 1 to level 6.
** **	Defines a top-level section.
<pre>	Defines a preformatted code section.
** **	Inserts a line break.
<hr />	Inserts a horizontal line extending across the page.

Tags that Affect Text Formatting

<b>	Displays text in boldface.
<i>	Displays text in italics.
<u>	Displays underlined text.
<code>	Displays text in the font used for code.
****	Encloses a run of text whose style can be set as a unit.

Other Useful HTML Tags

<script>	Encloses JavaScript (usually appears in the header).
<a>	Defines a hyperlink controlled to the **href** attribute.
<img>	Inserts an image as specified by the **src** attribute.
<!-- . . . **-->**	Indicates a comment in the HTML code.

- Several of these tags require additional information in the form of ***attributes***.
- In HTML, an attribute consists of a name, an equal sign, and a value, usually enclosed in quotation marks. For example, the sample page contains the following hyperlink:

```
<a href="http://en.wikipedia.org/wiki/World_Wide_Web">
```

Using the **<script>** Tag

- The **<script>** tag is used to enclose JavaScript code. This tag should specify the attribute **type="text/javascript"**.
- You can invoke JavaScript code by including an **onload** attribute in the **<body>** tag, as shown in the following code.

```
<!DOCTYPE html>
<html>
<head>
<title>Hello World</title>
<script type="text/javascript">
function sayHello() { alert("hello, world"); }
</script>
</head>
<body onload="sayHello()">
</body>
</html>
```

Capturing JavaScript Output

- The program example on the preceding slide displays its output by calling the built-in function **`alert`**, which pops up a message window containing the message.
- Although it might seem more natural to call **`console.log`** instead, most browsers make the console log difficult to find. The console log is used primarily by JavaScript developers, and it would confuse casual users if it appeared unexpectedly.
- In many cases, what you would like is for the output to appear in the contents of the web page. To accomplish that goal, you need to learn a little about how web pages are represented inside the browser, which is described briefly on the next slide.

The Document Object Model

- When the browser reads a web page, it first translates the text of the web page into an internal data structure that is easier to manipulate under the control of a JavaScript program. That internal form is called the ***Document Object Model***, or ***DOM***.
- The DOM is structured into a hierarchy of data objects called ***elements***, which usually correspond to a paired set of tags.
- The relationship between the HTML file and the internal representation is similar to the one between the external data file and the internal data structure in any data-driven program. The browser acts as a driver that translates the HTML into an internal form and then displays the corresponding page.
- Unfortunately, the DOM is extremely poorly designed, giving rise to a structure that is difficult to understand. The best strategy is to learn only those parts of the DOM you need.

Writing Text to a `<div>` Element

- The strategy we'll use in today's examples uses only two features of the DOM, both of which are reasonably simple:
  - *Naming an element in the HTML file.* It is often necessary to refer to a specific element in the web page from inside the JavaScript code. To do so, you need to include an **`id`** attribute in the HTML tag for that element that gives that element a name. JavaScript code can then find that element by calling **`document.getElementById(`***id***`)`**.
  - *Adding HTML content to an existing element.* The HTML code inside an element is available by selecting the **`innerHTML`** field of the element. The result is a JavaScript string that you can examine and modify.
- The code on the next slide writes the string **`"hello, world"`** into the **`<div>`** element whose **`id`** attribute is **`"log"`**.

An Improved Version of HelloWorld

```
<!DOCTYPE html>
<html>
<head>
<title>Hello World</title>
<script type="text/javascript">
function sayHello() {
   var div = document.getElementById("log");
   div.innerHTML = "hello, world";
}
</script>
</head>
<body onload="sayHello()">
<div id="log">
<!-- This is where the text eventually goes -->
</div>
</body>
</html>
```

Simulating a Countdown

```
<!DOCTYPE html>
<html>
<head>
<title>Hello World</title>
<script type="text/javascript">
function countdown(n) {
   for (var i = n; i >= 0; i--) {
      log(i);
   }
}

function log(str) {
   var div = document.getElementById("log");
   div.innerHTML += str + "<br />";
}
</script>
</head>
<body onload="countdown(10)">
<div id="log"></div>
</body>
</html>
```

Exercise: Factorial Table

Factorial Table

0! = 1
1! = 1
2! = 2
3! = 6
4! = 24
5! = 120
6! = 720
7! = 5040
8! = 40320
9! = 362880
10! = 3628800