HTML and CSS

Databases and Interfaces

Matthew Pike & Yuan Yao

University of Nottingham Ningbo China (UNNC)

Overview

This Lecture

- How to use HTML and CSS to create web pages?
- What tools do we need?
- How should we organise and structure our code?
- By the end of this lecture you should be able to create a simple web page using HTML and CSS

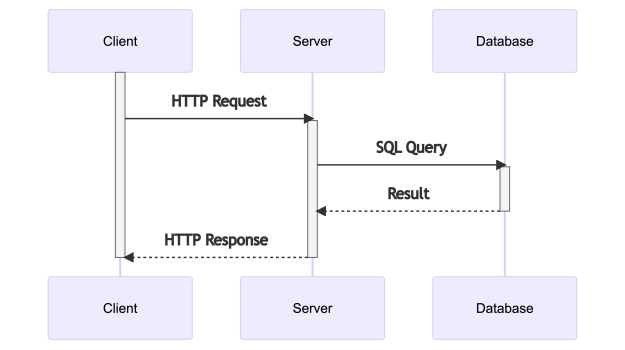
Client-Server Model

Hyper Text Transfer Protocol (HTTP)

- HTTP is a protocol for transferring data between a client and a server
- HTTP is a stateless, text-based protocol
  - The server does not remember the client between requests
- HTTP is a request-response protocol
  - The client sends a request to the server
  - The server sends a response to the client
- A server may return a response with a status code
  - 200: OK
  - 404: Not Found
  - 500: Internal Server Error
  - 503: Service Unavailable (Moodle's Favorite)

Client - Server Model

- The client is the user's web browser
- The server is a computer that hosts a website



The client-server model showing the request-response cycle. The figure also demonstrates the relationship between the database and the interface.

HTML

What makes a web page?

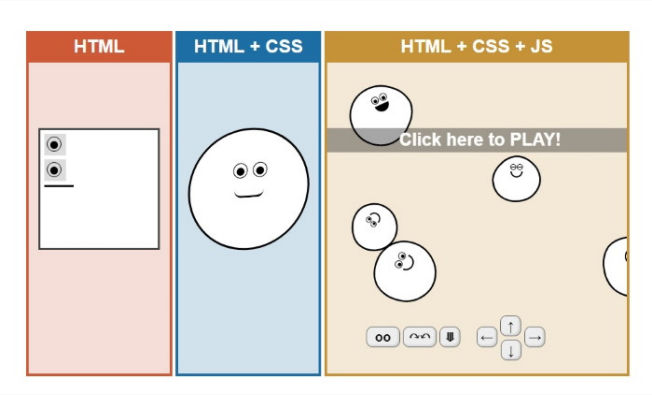


Figure 1: Image Source:html-css-js.com

HTML: Structure

CSS: Style

JavaScript: Interactivity

What is HTML?

Different Versions of HTML

In this module we will be using HTML5, the latest version of HTML.

- HTML stands for Hyper Text Markup Language
- HTML is a markup language used to create web pages using a series of elements
  - Markup languages are used to annotate text
  - Markup languages are not programming languages
- HTML elements allow you to structure your content using tags (examples to follow)
- The HTML language specification is maintained by the World Wide Web Consortium (W3C)
  - HTML 5.2 Specification is available at: https://www.w3.org/TR/html52/

HTML Tags

- HTML tags are used to define the structure of a web page
- HTML tags are enclosed in angle brackets and are case-insensitive
- HTML tags are divided into opening and closing tags
  - Opening tags are used to start an element
  - Closing tags are used to end an element
  - Opening and closing tags must be matched

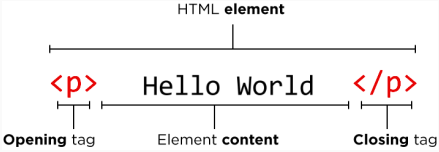


Figure 2: HTML Tags

Example: Hello World

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>Hello World</title>
</head>
<body>
    Hello World!
</body>
</html>
```

HTML Tags Hierarchy

- `<!DOCTYPE html>`: is the document type declaration. It tells the browser that the document is an HTML5 document
- `<html>`: is the root element of an HTML page and contains all other HTML elements
- `<head>`: contains metadata about the document. Content is not displayed
  - `<title>`: Sets the title of the document, shown in the browser's title bar
  - `<meta>`: Specifies metadata about the HTML document
    - `charset="utf-8"` specifies that the document is encoded using UTF-8
- `<body>` contains the visible page content
  - The body of an HTML document is the part that is displayed in the browser window

Nested HTML Tags

Web Browsers and Invalid HTML Code

Web browsers are very forgiving of invalid HTML code. They will try to display the content of the page even if the HTML is not valid. Nevertheless, it is important to ensure that your HTML is valid.

- HTML elements can be nested
  - Nesting means that one element is inside another element
  - The outer element is called the parent element
  - The inner element is called the child element
- Care must be taken to ensure that the opening and closing tags are matched correctly
  - Correct: `<p>Hello <b>COMP</b>1048!</p>`
  - Incorrect: `<p>Hello <b>COMP1048.</p></b>`

HTML Attributes

- HTML elements can have attributes which provide additional information about an element
- Attributes are always specified in the opening tag and usually come in name/value pairs like: `name="value"`
- Common attributes include:
  - `id`: Specifies a unique id for an HTML element. The `id` attribute given to an element must be unique within the HTML document
  - `class`: Specifies one or more class names for an HTML element. Multiple html elements can share the same class name.
- Examples:
  - `<p class="center">Hello World!</p>`
  - `<p id="intro">Hello World!</p>`
  - `<p class="center bold">Hello World!</p>`
- Later in the lecture we will see how to use CSS to style HTML elements using the `class` and `id` attributes

Headings, Paragraphs and Line Breaks

- HTML headings are defined with the `<h1>` to `<h6>` tags
  - `<h1>` defines the most important heading
  - `<h2>`, `<h3>`, `<h4>`, `<h5>` are less important headings
  - `<h6>` defines the least important heading
- HTML paragraphs are defined with the `<p>` tag
  - The browser automatically adds some white space (a margin) before and after a paragraph
- HTML line breaks are defined with the `<br>` tag
- Lists are defined with the `<ul>` or `<ol>` tags
  - The `<ul>` tag defines an unordered list, denoted by bullet points
  - The `<ol>` tag defines an ordered list, denoted by numbers
  - The `<li>` tag defines a list item for an unordered or ordered list

Example: Headings, Paragraphs and Line Breaks

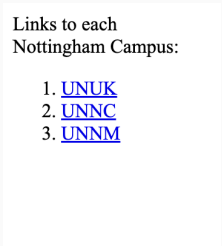
```
<!-- Header skipped for brevity -->
    <h1> Welcome! </h1>
    <p> This is <br /> my website. </p>
    <h2>Hobbies</h2>
    <ul>
        <li>Coding</li>
        <li>Playing Football</li>
    </ul>
    <h2> Football Teams </h2>
    <ol>
        <li>Swansea City</li>
        <li>Swansea City</li>
    </ol>
```

Hyperlinks

- HTML links are defined with the `<a>` tag
  - The `<a>` tag defines a hyperlink, which is used to link from one page to another
  - The most important attribute of the `<a>` element is the `href` attribute, which indicates the link's destination
  - The link text is the content between the opening and closing `<a>` tags
- The `href` attribute can be used to link to other web pages, files, email addresses, or any other URL
- Example:
  - `<a href="link-to-resource-goes-here">Link Text</a>`

Example: Hyperlinks

```
<!-- Header skipped for brevity -->
<p>Links to each Nottingham Campus:</p>
<ol>
    <li><a href="https://nottingham.ac.uk">
        UNUK
    </a></li>
    <li><a href="https://nottingham.edu.cn">
        UNNC
    </a></li>
    <li><a href="https://nottingham.edu.my">
        UNNM
    </a></li>
</ol>
```



Images

Closing Tags

We can often omit the closing tag if the element does not "contain" other tags or content. A shorthand for this is to use a forward slash (/) after the opening tag - `<br />` is equivalent to `<br></br>`.

- HTML images are defined with the `<img>` tag
- The `src` attribute is **required**, and contains the path to the image you want to embed.
- The `alt` attribute specifies an alternate text for the image, if the image for some reason cannot be displayed
  - Important for **accessibility**: Screen readers will read the `alt` text aloud to describe the image to visually impaired users
- `width` and `height` are also commonly used `image` attributes

Example: Images

```
<img
    src="images/missing.png"
    alt="A missing image"
    width="200"
    height="200"
/>

<img
    src="https://tinyurl.com/uonlogo"
    alt="University of Nottingham Logo"
/>
```

Tables

- HTML tables are defined with the `<table>` tag
  - `<tr>` defines a table row, `<th>` a table header and `<td>` a cell of data
- A table can be split into header (`<thead>`), data (`<tbody>`) and footer (`<tfoot>`) sections
- A caption can be added to a table with the `<caption>` tag
  - Again the caption can assist with accessibility, providing a description of the table for visually impaired users
- The `colspan` and `rowspan` attributes can be used to merge cells

Example: Tables

```
<table border=1>
<caption> DBI Class Schedule </caption>
<thead>
    <tr> <th>Week</th> <th>Topic</th> </tr>
</thead>
<tbody>
    <tr><td>1</td> <td>Introduction to DB</td> </tr>
    <tr> <td colspan="2"> <b> Holiday </b> </td> </tr>
</tbody>
<tfoot>
    <tr> <td> Updated </td> <td>08 November 2022</td>
</tfoot>
</table>
```

DBI Class Schedule

Week	Topic
1	Introduction to DB
Holiday	
Updated	08 November 2022

Forms

- The `<form>` tag defines an interactive control for user input
- Clicking on the submit button will send the data to the URL specified `action` attribute.
- Data is sent to the server as name/value pairs, where the name is defined by the `name` attribute
- The `method` attribute defines how to send the form-data (GET or POST)
- The `<input>` tag defines an input control, with the `type` attribute defining the type of input control to display
  - Examples of input `type`: `text`, `password` and `submit`
- The `<label>` tag defines a label for an `<input>` element. The `for` attribute of the `<label>` tag should be equal to the `id` attribute of the `<input>` element to bind them together

Example: Forms

```
<form action="example.com/process_form" method="post">
    <label for="frmEmail">Email:</label>
    <br>
    <input type="text" id="frmEmail"
        name="email"
        value="test@email.com">
    <br/>
    <label for="frmPswd">Password:</label>
    <br>
    <input type="password" id="frmPswd"
        name="password" value="password">
    <br/><br/>
    <input type="submit" value="Submit">
</form>
```

GET vs POST

- The `GET` method is used to retrieve information from the given server using a given URI
  - Requests using `GET` should only retrieve data
  - `GET` requests should not be used to send (especially sensitive) data, because the sent data is visible in the URL
  - `GET` requests are limited in the amount of data they can send
- The `POST` method is used to send data to the server, for example, customer information, file upload, etc.
  - The `POST` method allows for sending larger amounts of data than `GET`
  - `POST` is also safer and more robust than `GET` because the parameters are not stored in browser history or in web server logs
  - `POST` request data are sent in the message body of the HTTP request, with the header `Content-Type: application/x-www-form-urlencoded` or `multipart/form-data`

Practical Hints and Tips for developing HTML

- Ensure that your HTML is valid
  - Use the W3C Markup Validation Service to check your HTML - validator.w3.org
- Use a text editor with syntax highlighting
  - Visual Studio Code is a good choice
- Use a *good* web browser
  - Google Chrome
  - Firefox
- Learn to use the browser developer tools
  - Chrome DevTools - developers.google.com/web/tools/chrome-devtools
  - Firefox Developer Tools - developer.mozilla.org/en-US/docs/Tools
- Use the Mozilla Developer Network (MDN) as a reference:
  - developer.mozilla.org/en-US/docs/Web/HTML/

CSS: Cascading Style Sheets

What is CSS?

- Using CSS we can control exactly how HTML elements look in the browser, instead of relying on our browser's default styling
- Therefore, we can say:
  - HTML describes the structure of Web pages
  - CSS describes the style and presentation of Web pages
- CSS provides a diverse range of functionality, including:
  - Simple text formatting and layout
  - Complex multi-column layouts and responsive design
  - Animations and transitions
  - Fonts and typography

Recall: What makes a web page?

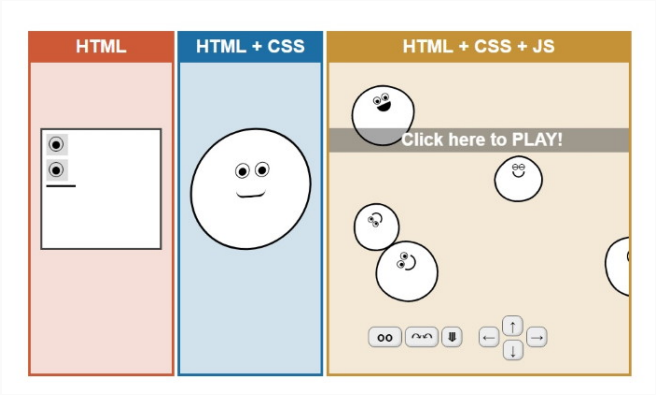


Figure 3: Image Source:html-css-js.com

CSS Syntax

CSS uses American English spelling, so `color` is used instead of `colour`.

- CSS is a rule based language
- The *selector* points to the HTML element you want to style
- The declaration block is wrapped in curly braces `{ }` and contains one or more declarations separated by semicolons `;`
- Each declaration includes a CSS property name and a value, separated by a colon `:`

Syntax:

```
selector {
    property: value;
    property: value;
}
```

Example:

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

CSS Selectors

- CSS selectors are used to “find” the HTML elements you want to style
- Element selectors: Selects elements based on the element name
  - Example: `p` selects all `<p>` elements
- Class selectors: Selects elements based on their `class` attribute using a `.` prefix
  - Example: `.center` selects all elements with `class="center"`
- ID selectors: Selects a single element based on the value of it’s `id` attribute using a `#` prefix
  - Example: `#intro` selects the element with `id="intro"`

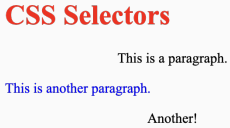
Example: CSS Selectors

HTML

```
<h1> CSS Selectors </h1>
<p class="center">
    This is a paragraph.
</p>
<p id="myPara">
    This is another paragraph.
</p>
<p class="center"> Another! </p>
```

CSS

```
h1 {color: red;}
.center {text-align: center;}
#myPara {color: blue;}
```



CSS Property Values

- CSS property values can be specified in a number of different ways:
  - Keywords: `center`, `left`, `right`, `top`, `bottom`, `red`, `blue`, `green`, etc.
  - Length units:
    - Pixels: `10px`, `20px`, `30px`, etc.
    - Percentages: `10%`, `20%`, `30%`, etc.
    - Centimeters: `10cm`, `20cm`, `30cm`, etc.
  - Colors:
    - Named colors: `red`, `blue`, `green`, etc.
    - Hexadecimal: `#ff0000`, `#00ff00`, `#0000ff`, etc.
    - RGB: `rgb(255, 0, 0)`, `rgb(0, 255, 0)`, `rgb(0, 0, 255)`, etc.

Adding CSS to HTML

- There are three ways to add CSS to HTML:
  - **Inline CSS**: Add CSS directly to the HTML element using the `style` attribute
  - **Internal CSS**: Add CSS to the `<head>` section of the HTML document using the `<style>` element
  - **External CSS**: Add CSS to a separate file and link to it in the `<head>` section of the HTML document using the `<link>` element
- You should use external CSS files, as this separates your content from it's styling
  - This makes it easier to maintain your code
  - It also allows you to reuse the same CSS file across multiple HTML documents
  - We link to an external CSS file using the `<link>` element in the `<head>` section of the HTML document
  - Example: `<link rel="stylesheet" href="style.css">`

Cascade Order of CSS Rules

- The order in which CSS rules are applied is called the *cascade order*
- The cascade order is determined by:
  - **Specificity**: The more specific a selector is, the higher the priority
  - **Source order**: The order in which the CSS rules appear in the CSS file
  - **Last rule**: If two rules have the same specificity, the last rule will take precedence

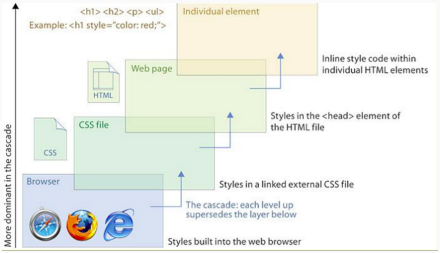


Figure 4: Image Source: Basic Design Principles for Creating Web Sites, by Patrick J. Lynch and Sarah Horton

The Box Model

- The box model is a way of thinking about HTML elements
- It describes how elements are laid out on the page
- It is made up of:
  - Content: The content of the element, such as text or images
  - Padding: Clears an area around the content. The padding is transparent
  - Border: A border that goes around the padding and content
  - Margin: Clears an area outside the border. The margin is transparent

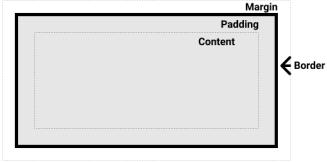


Figure 5: Image Source: MDN: The Box Model

How Does CSS Work?

1. The browser reads the HTML document and builds a tree of nodes called the Document Object Model (DOM)
2. The browser fetches other resources that are linked to the HTML document (e.g. images, CSS files, etc.)
3. The browser reads the CSS files and calculates the final style rules for each node in the DOM tree
4. The browser displays the HTML document, using the style rules to determine how to display the nodes in the DOM tree

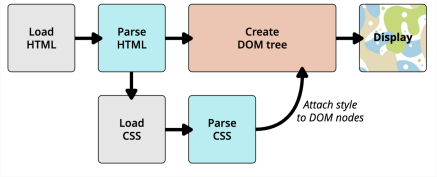


Figure 6: Image Source: MDN Web Docs: How CSS Works

The Document Object Model (DOM)

- The Document Object Model (DOM) is an in-memory representation of the HTML document used by the browser when rendering the page
- The DOM is a tree of objects, where each object represents a part of the document
- The DOM can be modified using JavaScript
- We can inspect the DOM using the browser's developer tools
  - In Chrome: Press `F12` → `Elements` tab
  - In Firefox: Press `F12` → `Inspector` tab
  - In Safari: Press `Option+Command+i` → `Elements` tab

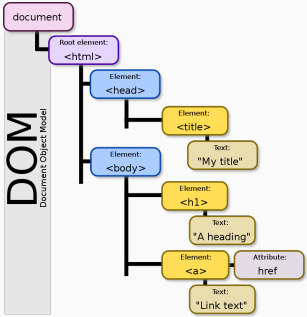


Figure 7: Image Source: Wikipedia: DOM

Answers to common questions

- We do not expect you to memorise all of the HTML and CSS tags and properties - this is obviously unreasonable
- We will expect you to be able to:
  - Read HTML and CSS code and understand what it does
  - Write simple HTML and CSS code using the tags and properties we have covered
- We want you to be able to create simple web pages using HTML and CSS and to be able to use the developer tools to inspect and debug your code

Resources

- HTML Validator: validator.w3.org/
- CSS Validator: jigsaw.w3.org/css-validator/
- MDN HTML Tutorial: developer.mozilla.org/en-US/docs/Learn/HTML
- MDN CSS Tutorial: developer.mozilla.org/en-US/docs/Learn/CSS
- CSS Zen Garden: csszengarden.com/

References and Further Reading

- MDN HTML Web Docs: developer.mozilla.org/en-US/docs/Web/HTML
- MDN CSS Web Docs: developer.mozilla.org/en-US/docs/Web/CSS
- HTML Specificationhtml.spec.whatwg.org/multipage/
- CSS Specificationw3.org/Style/CSS/specs.en.html