HTML and CSS

Databases and Interfaces

Matthew Pike & Yuan Yao

University of Nottingham Ningbo China (UNNC)

Overview

This Lecture

- How to utilise HTML and CSS for web page development.
- The essential tools for web development.
- Effective organisation and structuring of code.
- By the end of this lecture, you should be capable of creating a basic web page with HTML and CSS.

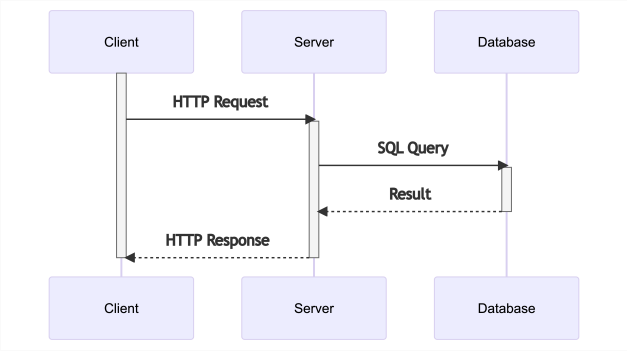
Client-Server Model

Hyper Text Transfer Protocol (HTTP)

- HTTP is the protocol used to transfer data between a client and a server.
- HTTP operates as a stateless, text-based protocol.
  - The server does not retain any memory of the client for subsequent requests.
- HTTP follows a request-response model.
  - The client issues a request to the server.
  - The server then dispatches a response back to the client.
- Servers respond with various status codes, such as:
  - 200: OK
  - 404: Not Found
  - 500: Internal Server Error
  - 503: Service Unavailable (often seen with Moodle)

Client - Server Model

- The client refers to the user's web browser.
- The server is a computer system that serves the hosted website's content and functionality.



This diagram illustrates the client-server model, highlighting the request-response cycle and the interactions with a database.

HTML

What makes a web page?

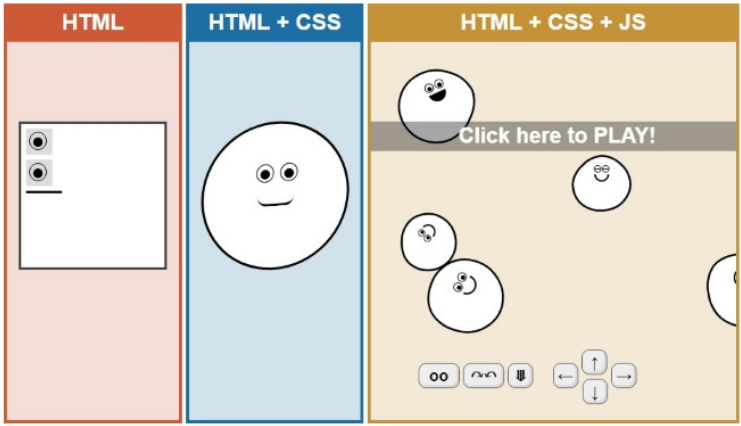


Figure 1: Image Source:html-css-js.com

What is HTML?

> **DBI uses HTML5 (only)**
>
> We will be using HTML5 exclusively in this module. The HTML 5 specification is maintained by the World Wide Web Consortium (W3C) - http://www.w3.org/TR/html5/

- Hyper Text Markup Language (HTML) is the markup language for creating web pages.
  - It's a markup language that annotates text, providing structural information to documents. It is not a programming language.
- HTML elements enable the structuring of content into headings, paragraphs, lists, tables, forms, and more. Common elements include `<h1>`-`<h6>`, `<p>`, `<ul>`/`<ol>`/`<li>`, `<table>`, and `<form>`. These will be explored in further detail later.
- Notable enhancements in HTML5 include new semantic elements such as `<header>`, `<footer>`, `<nav>`, and built-in multimedia support with `<video>` and `<audio>`.

HTML Tags

- HTML tags are the fundamental components used to construct web pages.
- These tags, which are case-insensitive, are contained within angle brackets: `<tag>` for the opening and `</tag>` for the closing.
- Tags usually appear in pairs, consisting of an opening tag and a corresponding closing tag: `<p>Hello World!</p>`.

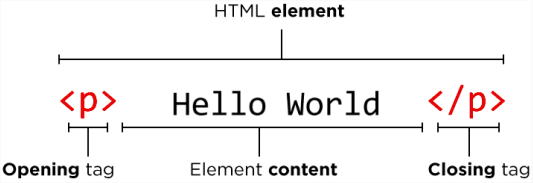


Figure 2: An annotated example of HTML tags, which include the opening and closing tags, the content, and the element

Example: Hello World

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

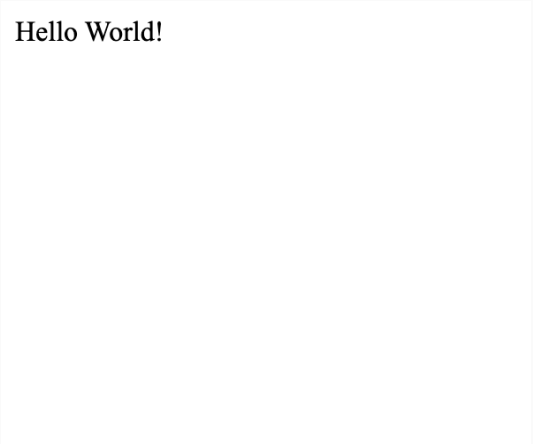


Figure 3: The rendered HTML page for a simple Hello World example.

HTML Tags Hierarchy

- `<!DOCTYPE html>` declares the document type and version of HTML.
- `<html>` serves as the root of the HTML document, encompassing all content except the `<!DOCTYPE html>`.
- `<head>` holds the metadata of the document, which doesn't appear on the display.
  - `<title>` defines the document's title, which appears in the browser's title bar.
  - `<meta>` details the document's metadata.
    - `charset="utf-8"` denotes that the document uses UTF-8 encoding.
- `<body>` encapsulates the content visible on the webpage.
  - The body element contains what is rendered in the browser's viewing area.

Nested HTML Tags

Web Browsers and Invalid HTML Code

Web browsers are highly tolerant of invalid HTML code and will attempt to render the page content even when the HTML is not valid. Nonetheless, it is crucial to verify 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>`
- There are many tools available to help you check that your HTML is valid
  - W3C Markup Validation Service - validator.w3.org

HTML Attributes

- HTML elements can have attributes that provide additional details about the element.
- Attributes are always placed in the opening tag and typically follow a name/value pattern like: `name="value"`.
- Common attributes include:
  - `id`: Assigns a unique id to an HTML element. Each `id` attribute must be unique within the HTML document.
  - `class`: Allocates one or more class names to an HTML element. The same class name can be used by multiple elements.
- Examples:
  - `<p class="center">Hello World!</p>`
  - `<p id="intro">Hello World!</p>`
  - `<p class="center bold">Hello World!</p>`
- We will explore how to use CSS to style HTML elements using `class` and `id` attributes later in the lecture.

Headings, Paragraphs and Line Breaks

- HTML headings are defined with the `<h1>` to `<h6>` tags.
  - `<h1>` denotes the most significant heading.
  - `<h2>`, `<h3>`, `<h4>`, `<h5>` signify progressively less important headings.
  - `<h6>` represents the least significant heading.
- HTML paragraphs are marked with the `<p>` tag.
  - The browser automatically applies some white space (a margin) before and after a paragraph.
- HTML line breaks are indicated with the `<br/>` tag.
- Lists are created using the `<ul>` or `<ol>` tags.
  - `<ul>` defines an unordered list, represented by bullet points.
  - `<ol>` specifies an ordered list, indicated by numbers.
  - `<li>` is used for each item in either an unordered or an 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>
```

Figure 4: The rendered HTML page.

Hyperlinks

- HTML links are denoted with the `<a>` tag.
  - The `<a>` tag defines a hyperlink, utilised for linking from one page to another.
  - The main attribute of the `<a>` element is the `href`, which specifies the link's target.
  - The link text is what appears between the opening and closing `<a>` tags.
- The `href` attribute is versatile, allowing links to other web pages, files, email addresses, or any URL.
- Example:
  - `<a href="http://www.nottingham.edu.cn">Visit our homepage</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>
```

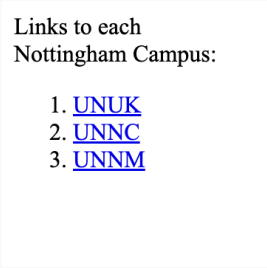


Figure 5: The rendered HTML page for the hyperlinks example.

Images

Closing Tags

Closing tags can typically be omitted if the element does not enclose other tags or content. Using a forward slash (`/`) at the end of the opening tag is a shorthand for this – `<br />` is the same as `<br></br>`.

- HTML images are inserted using the `<img>` tag.
- The `src` attribute, which is **mandatory**, holds the path to the desired image.
- The `alt` attribute provides alternative text for the image should it fail to display.
  - This is essential for **accessibility**, as screen readers convey the `alt` text to users with visual impairments.
- The attributes `width` and `height` are frequently employed to set the dimensions of the image.

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"
/>
```

Figure 6: The rendered HTML page for the images example.

Tables

> ⚠ Do not use tables for layout
>
> Tables should not be used for layout purposes. Instead, use CSS for layout. This is a common mistake made by beginners.

- HTML tables are created with the `<table>` tag.
  - `<tr>` specifies a table row, `<th>` denotes a table header, and `<td>` represents a data cell.
- A table can be divided into header (`<thead>`), body (`<tbody>`), and footer (`<tfoot>`) sections.
- A caption can be included using the `<caption>` tag.
  - The caption also supports accessibility by offering a description of the table's contents for users with visual impairments.
- The attributes `colspan` and `rowspan` allow for the merging of 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> </tr>
</tfoot>
</table>
```

DBI Class Schedule

Week	Topic
1	Introduction to DB
Holiday	
Updated	08 November 2022

Figure 7: The rendered HTML page for the tables example.

Using `<div>` and `<span>`

- The `<div>` tag defines a block-level element to group other elements, commonly used for organising elements into logical sections.
- The `<span>` tag is an inline element for grouping within other elements, typically used to apply styles to text sections within a paragraph.
- `class` and `id` attributes are used to associate styles (via CSS) to `<div>` and `<span>` elements.

Forms

- The `<form>` tag establishes an interactive form for user input.
- Submitting the form sends the data to the server, targeting the URL specified in the `action` attribute.
- Data is dispatched as name/value pairs, with 'name' being designated by the `name` attribute.
- The `method` attribute specifies the data submission method (either GET or POST).
- The `<input>` tag creates an input field, where the `type` attribute determines the kind of input field presented.
  - Varieties of input `type` include: `text`, `password`, and `submit`.
- The `<label>` tag labels an `<input>` element. The `for` attribute of `<label>` should match the `id` attribute of `<input>` to link them.

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">
```

Figure 8: The rendered HTML page for the forms example.

GET vs POST

- The `GET` method retrieves information from a server at a specified URI.
  - `GET` should only be used to retrieve data.
  - `GET` is not suitable for submitting sensitive data, as the parameters are visible in the URL.
  - `GET` requests have limitations on data length.
- The `POST` method sends data to the server, such as customer information or file uploads.
  - `POST` can send more data compared to `GET`.
  - `POST` is more secure and robust, as the parameters are not saved in browser history or web server logs.
  - Data in `POST` requests are embedded in the HTTP request's body, with headers like `Content-Type: application/x-www-form-urlencoded` or `multipart/form-data`.

Practical Hints and Tips for developing HTML

- Ensure that your HTML is valid.
  - Validate your HTML using the W3C Markup Validation Service - validator.w3.org.
- Utilise a text editor with syntax highlighting.
  - Visual Studio Code is an excellent option.
- Use a *good* web browser.
  - Google Chrome
  - Firefox
- Familiarise yourself with 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) for reference:
  - developer.mozilla.org/en-US/docs/Web/HTML/

CSS: Cascading Style Sheets

What is CSS?

- CSS enables us to specify the appearance of HTML elements in the browser, beyond the default styles provided by the browser.
- Thus, it is accurate to state:
  - HTML defines the structure of web pages.
  - CSS dictates the style and presentation of web pages.
- CSS offers an array of features, including:
  - Basic text formatting and layout management.
  - Complex multi-column arrangements and responsive design.
  - Implementation of animations and transitions.
  - Fonts and typographic details.

Recall: What makes a web page?

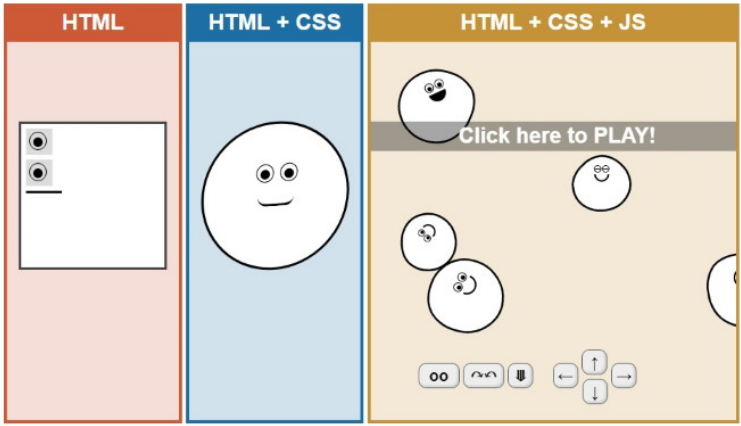


Figure 9: Image Source:html-css-js.com

CSS Syntax

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

- CSS is a rule-based language.
- The *selector* targets the HTML element to be styled.
- The declaration block, enclosed in curly braces `{}`, houses one or more declarations, each separated by a semicolon `;`.
- Each declaration consists of 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 identify the HTML elements to apply styles to.
- Element selectors: Choose elements by their tag name.
  - Example: `p` will target all `<p>` elements.
- Class selectors: Select elements with a specific `class` attribute, prefixed by a `.`.
  - Example: `.center` will select all elements with `class="center"`.
- ID selectors: Target a unique element based on its `id` attribute value, prefixed by a `#`.
  - Example: `#intro` will select the element with `id="intro"`.

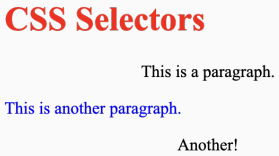
Example: CSS Selectors

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 defined in various formats:
  - Keywords: `center`, `left`, `right`, `top`, `bottom`, 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 codes:`#ff0000`, `#00ff00`, `#0000ff`, etc.
    - RGB values: `rgb(255, 0, 0)`, `rgb(0, 255, 0)`, `rgb(0, 0, 255)`, etc.

Adding CSS to HTML

- There are three methods to apply your CSS to your HTML:
  - **Inline CSS**: Embed CSS directly into an HTML element using the `style` attribute. Avoid.
  - **Internal CSS**: Place CSS within the `<head>` portion of the HTML document using the `<style>` tag. Not recommended.
  - **External CSS**: Create CSS in a separate file and reference it within the `<head>` section of the HTML document using the `<link>` tag. Recommended.
- It's advisable to use external CSS files to decouple your content from its styling.
  - This approach simplifies code maintenance.
  - It facilitates the reuse of the same CSS file in various HTML documents.
  - We link an external CSS file using the `<link>` tag in the `<head>` part of the HTML document.
  - Example: `<link rel="stylesheet" href="styles.css">`

Cascade Order of CSS Rules

- The sequence in which CSS rules are executed is known as the *cascade order*.
- The cascade order is influenced by:
  - **Specificity**: Selectors with greater specificity have higher priority.
  - **Source order**: The sequence in which the CSS rules are listed in the CSS file.
  - **Last rule**: When two rules have equal specificity, the latter rule prevails.

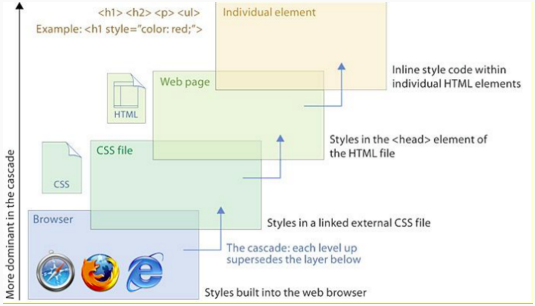


Figure 11: Image Source: Basic Design Principles for Creating Web Sites, by Patrick J. Lynch and Sarah Horton

The Box Model

- The box model is a conceptual framework for HTML elements.
- It explains the placement and layout of elements on the page.
- It consists of:
  - Content: The actual content of the element, like text or images.
  - Padding: Space that clears an area around the content; padding is transparent.
  - Border: A border that encircles both the padding and the content.
  - Margin: Space that clears an area outside the border; the margin is transparent.

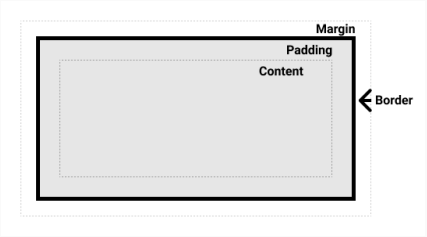
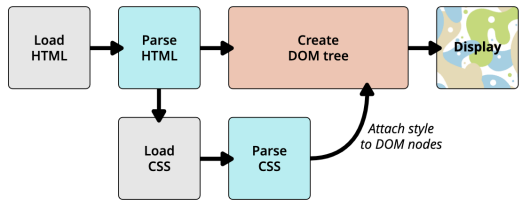


Figure 12: Image Source: MDN: The Box Model

How Does CSS Work?

1. The browser interprets the HTML document and constructs a tree structure known as the Document Object Model (DOM).
2. It then retrieves additional resources linked to the HTML document, such as images and CSS files.
3. The browser processes the CSS files to establish the definitive style rules applicable to each node in the DOM tree.
4. Finally, the browser renders the HTML document on the screen, applying the style rules to dictate the presentation of nodes within the DOM tree.



The Document Object Model (DOM)

- The Document Object Model (DOM) is a browser's in-memory representation of the HTML document for page rendering.
- The DOM is structured as a tree of objects, with each object corresponding to a part of the document.
- JavaScript can be used to alter the DOM - something we will explore in a later lecture.
- The DOM is accessible through the browser's developer tools:
  - In Chrome: Press `F12` and select the `Elements` tab.
  - In Firefox: Press `F12` and go to the `Inspector` tab.
  - In Safari: Press `Option+Command+i` and click on the `Elements` tab.

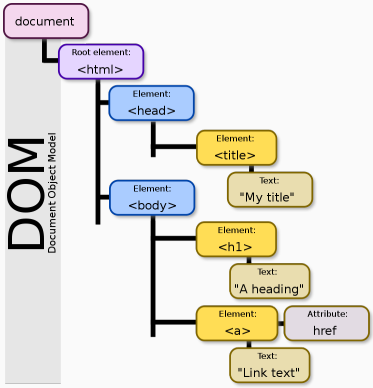


Figure 14: Image Source: Wikipedia: DOM

Answers to common questions

- It’s not expected for you to memorise all HTML and CSS tags and properties, as this is clearly impractical.
- You should be able to:
  - Interpret HTML and CSS code and comprehend its function.
  - Create basic HTML and CSS code using the tags and properties we’ve discussed.
- The goal is for you to construct simple web pages using HTML and CSS, and utilise developer tools to inspect and troubleshoot 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 Specification: html.spec.whatwg.org/multipage
- CSS Specification: w3.org/Style/CSS/specs.en.html