Transcript

Module 3: HTML and CSS Fundamentals

Video 1: Module Overview

Welcome to the "HTML and CSS Fundamentals" module. This module is designed to give you a strong foundation in building and styling web pages. Let's dive into the key topics and what you will learn.

First, we'll explore the Structure & Syntax of HTML pages. HTML, or HyperText Markup Language, is the backbone of web pages. Understanding its structure and syntax is crucial as it dictates how content is organized and displayed. You'll learn about elements, tags, attributes, and how to construct a well-formed HTML document.

Next, we'll cover Common HTML tags. These are the building blocks of web content, including headings, paragraphs, lists, links, images, and more. Mastering these tags will enable you to create rich and structured web content.

We'll then delve into Working with URLs. URLs, or Uniform Resource Locators, are addresses used to access resources on the web. You'll understand the different types of URLs, when to use absolute URLs, and when to use relative URLs. This knowledge is essential for linking to other pages and resources correctly.

Following that, we'll focus on Working with Form-Related Tags. Forms are a crucial part of web interaction, allowing users to submit data. You'll learn about various form elements like input fields, checkboxes, radio buttons, and how to process form data.

Next, we'll introduce CSS (Cascading Style Sheets). CSS is used to style HTML content, making it visually appealing. You'll learn the basics of writing CSS rules and the three ways to declare CSS: inline, internal, and external styles.

Understanding the Document Object Model (DOM) is also essential. The DOM represents the structure of a web page and allows for dynamic content manipulation. You'll learn how to use browser developer tools to inspect and modify DOM elements, enhancing your debugging and styling skills.

We'll also cover Basic CSS Selectors. Selectors are used to target HTML elements for styling. You'll learn about different types of selectors, including element, class, and ID selectors, and how to use them effectively.

In addition to selectors, you'll explore Common CSS

Properties such as color, margin, padding, and layout.

These properties help you control the appearance and spacing of elements on your web pages.

By the end of this module, you'll have a solid understanding of HTML and CSS, enabling you to create and style web pages effectively. You'll be well-prepared to continue your web development journey and tackle more advanced topics.

Video 2: Structure & Syntax of HTML pages

Welcome to the lecture on "Structure & Syntax of HTML pages." Here are the learning objectives. By the end of this lecture, you should be able to understand the structure and syntax of HTML pages.

HTML stands for HyperText Markup Language. It is a scripting language used for creating web pages. Essentially, HTML is a text document that defines the format, layout, and resources in a web page.

Getting The HTML Source: To understand HTML better, let's see how we can view the HTML source of a webpage. By right-clicking on a webpage and selecting 'View Page Source', you can see the HTML code that is used to construct the page.

Structure Of An HTML Page: An HTML page consists of a DOCTYPE declaration, html tag, head tag and body tag. The following is an example of a basic HTML document. DOCTYPE defines the document type and version of HTML used. The html tag is the root of the HTML document. Each webpage typically consists of a head section and a body section.

DOCTYPE Tag: The DOCTYPE defines the type of the document, for instance, XHTML 1.0 uses the following DOCTYPE.

While HTML5 simply uses the following.

Specifying DOCTYPE while not necessary is useful to help the web browser better interpret the document more correctly.

HTML Tagging Structure: HTML tags usually come in pairs, with a starting and an ending tag. Here is an example.

Some tags are self-closing, typically void elements as shown here.

However, for non-void elements, you should always use a closing tag.

HTML Tagging Structure (Attributes): Tags can have attributes that provide additional information. Attributes are enclosed in quotation marks using either the double quotations or single quotations marks. Just ensure that you use matching quotation marks.

Attributes can also be boolean, where the presence of the attribute implies its value is true.

HTML Comments: Like in coding, comments in HTML are used to add information to the developer without displaying them on the page. Comments are helpful for explaining your code or temporarily disabling parts of your HTML during development.

Now let's dive into the head-related tags. The head section contains information about the page. Elements defined here are usually not visible to users, except for the title tag. The commonly used tags in this section include the title, meta, style, link, and script tag. Among which, the style, link and script tags can also be placed in the body section.

The title tag defines the title of the page, which is displayed in the browser's title bar or tab. Here's an example.

As you can see from the image, the title is displayed at the top of the browser window.

The meta tag is used to capture various meta-information about the page, such as descriptions, keywords, and author information. These tags are traditionally used by search engines for indexing pages. In the code example here, the viewport meta tag is particularly useful for mobile devices, ensuring the page is more readable.

In addition to search engines, meta tags are now used by social media sites to enhance how your shared content is displayed on the platforms.

The style and link tags are used to define CSS, which controls the style of the page. These tags can be placed in either the head or body section but are usually found in the head. Here is an example of how they are written.

The script tag is used to include JavaScript in your HTML. It can be placed in either the head or body section. Previously, the syntax included the type attribute.

With HTML5, the type attribute is no longer necessary.

You can also write JavaScript directly within the script tags.

<script> Pitfall: When linking an external Javascript, note that you should not use the self-closing void tag even though you are just trying to define the Javascript location. If you were to do this, you will realize that the page will not render properly.

Body-Related Tags: The body section contains the presentational content of the page. It can be divided into several categories, including: Text, Hyperlinks, (Tables, lists, and blocks), Images, Iframes, and Form elements.

To summarize, we have covered the basic structure and syntax of HTML pages. Understanding these concepts is fundamental to creating effective and well-structured web pages.

Video 3: Commonly Used HTML Tags

Welcome to the lecture on "Commonly Used HTML Tags". Here are the learning objectives. By the end of this video, you should understand how to use the common HTML tags. These tags form the building blocks of HTML and are essential for creating web pages.

This video will focus mainly on the body section tags as they focus on presenting content on web pages. Here are the categories of body tags we will be covering in this video.

These are some text-related tags that can be used. They are similar to what we see in a Microsoft Word document. Headings can be defined using h1, h2 to h6 where h1 is the biggest heading and h6 is the smallest heading. p tag is used to represent paragraphs. We can also apply some basic text styling such as bold, italic, underline, superscript, subscript. The blockquote tag can be used to define a quotation section.

When displaying text, note that multiple spaces will only be displayed as a single space, and new lines in the file have no effect unless you use the BR tag. Alternatively, you can use the PRE tag to preserve new line and space formatting written in the html document.

To get started with HTML, we would recommend using the Visual Studio Code or VS Code editor. Here’s a screenshot of Visual Studio Code. It is a versatile editor with many features that support HTML and CSS development. It is a powerful code editor that is very popular amongst the development community. You can download it from the visual studio code website as shown here.

Exercise: Create A Hello World Page: To help you get started with HTML and VS Code, try pausing this video and see whether you are able to create the following web page. You can resume this video once you are done. Here is one possible way how to write the HTML codes in order to achieve the expected output.

Links: To create hyperlinks, you can use the a or the anchor tag and define the HREF attribute for the URL. You can add the target="_blank" attribute to open the link in a new window. Remember, hyperlinks are not created using the link tag. The link tag is used to define the CSS of the page.

To lay out text, you can use lists, tables, and blocks. These elements help organize content and improve readability. The 2 commonly used lists in HTML is the ordered list and unordered list. Ordered lists are those where the list items are numbered while unordered lists are those where the list items are by default using bullet points. The lists are defined using the OL or UL tags, and the items in the lists are defined using LI.

Tables are defined within the table tags. They work similar to how we would define tables in a Microsoft Word document where a table is made up of rows and columns. We use TR to define a table row, and TD or TH to define columns within a row. Each row should have an equal number of columns, but you can merge columns using the COLSPAN attribute or merge rows using the ROWSPAN attribute.

To define sections in your page, use the DIV tag for block-level elements, and the SPAN tag for inline elements. Here's are some examples.

These tags are useful for styling and layout.

To add images to your page, use the I-M-G or image tag and define the S-R-C or source attribute for the image URL.

The IFRAME tag allows you to embed another webpage within your current page. It is commonly used to embed YouTube videos in a web page.

In summary, we've covered the commonly used HTML tags which including text, links, lists, tables, blocks, images, and iframes.

Understanding these tags is crucial for creating well-structured and functional web pages.

Video 4: Working with URLs

Welcome to the lecture on "Working with URLs". Here are the learning objectives. By the end of this video, you should understand the different types of URLs, when to use absolute URLs and relative URLs. These concepts are crucial for properly linking resources and navigation within web pages.

A URL, or Uniform Resource Locator, is used for locating resources on the web. URLs consist of several components:

Scheme specifies the protocol such as http, https, ftp.

Host is the domain name

Path is the location of the resource on the server.

We can also specify a query which is additional parameters. Finally, fragment is a specific section within the resource.

URLs are used in webpages primarily for two purposes. To specify the hyperlinks to other web pages, and to specify resources such as images, audio and video files. URLs are essential for web navigation and resource management.

There are two main types of URLs. Absolute URLs and Relative URLs.

Absolute URLs provide the full path to the resource, including the scheme and domain.

Relative URLs on the other hand, provide a path relative to the current document's location.

Both types have their specific use cases, which we will discuss next.

Absolute URLs should be used when linking to external websites.

Advantages of absolute URLs include that it is unambiguous and consistent. It also ensures that links to external sites remain functional even if your document's location changes.

Relative URLs are ideal for linking within the same website. They are used for internal links, images, stylesheets, and scripts.

Advantages of relative URLs include its simplicity, easier management during site migration and shorter path lengths.

We will discuss some examples of relative URLs. This is an example of the case where we want to link to a webpage in the same directory.

It is also possible to link to another subdirectory by specifying the directory name followed by the file we want in the subdirectory.

To access the immediate parent directory, we add two period in front of the URL.

Sometimes it might be easier to just access resources relative to the root directory. To do that, add a slash in front of the URL. This will then disregard the current location of this page.

In summary, we have covered the different types of URLs, when to use absolute URLs and relative URLs.

Understanding these concepts is essential for effectively managing web links and resources.

Video 5: Working with Form-Related Tags

Welcome to the lecture on "Working with Form-Related Tags". Here are the learning objectives. By the end of this lecture, you should understand how to work with form-related tags in HTML. Forms are a crucial part of web development as they allow for user input and interaction.

Form elements should be enclosed within the FORM tag. Typically, forms include 2 attributes. The ACTION and METHOD attributes. The ACTION attribute specifies the URL to which the form data will be submitted, and the METHOD attribute specifies the HTTP method to use, either GET or POST. Within a form, you can include various elements which we will discuss next. Each of these elements usually has the NAME and VALUE attributes to identify and capture user input.

Let's talk about the INPUT tag. The INPUT tag can represent different types of input fields by specifying different value for the TYPE attribute. Here are some common values for the TYPE attribute. We should be able to easily guess what type of form elements are generated based on the TYPE attribute value.

These include:

the TEXT value which create a textbox.

the PASSWORD value which generates a textbox where the contents are masked like what we see in a registration form.

the FILE value which popups a file selection dialog for users to select a file to upload, just to name a few.

Apart from the standard form elements, HTML5 has also introduced several new input types to cater for today's demand. The new type values include: URL, EMAIL, DATE, TIME, and COLOR. New attributes have also been added such as the AUTOCOMPLETE, PLACEHOLDER and REQUIRED attributes. These features were traditionally achieved using JavaScript.

Let's dive into some specific examples of these new HTML5 elements. The URL input type generates a textbox that focuses on getting the users to enter URLs. If the user does not enter a valid URL, the form will not be submitted, and the user will be prompted a note to enter a valid URL.

Similarly, the EMAIL type is used for email addresses.

If a textbox is supposed to capture date value, we could define the DATE type. Depending on the web browser, the web browser might render a date picker component for the users to select the date instead of having to manually type in the date value.

Similarly, the TIME type is used to capture the time information. Similar to the date picker component, in here a time picker component is rendered.

HTML5 also provide a color picker input type for selecting color values.

Next, let's talk about labels. It's important to add a LABEL for each form element. Labels improve accessibility and usability by linking text descriptions to form elements. What do you think a LABEL tag is used for other than displaying the text?

The main benefit of specifying a label is so that when users were to click on the label, the corresponding form elements will be highlighted.

You can use the AUTOFOCUS attribute to focus on a form field automatically when the page loads. Note that you should only use one autofocus element for each page.

HTML5 also provides simple validation attributes like the MIN and MAX attributes. These attributes help enforce input constraints without the use of JavaScript.

It is also possible to specify different kinds of validation rules using the MINLENGTH and MAXLENGTH attribute which specify the minimum and maximum number of characters for an input field.

In summary, we have covered the essential form-related tags in HTML.

Understanding these elements will help you create effective and user-friendly forms.

Video 6: Introduction to CSS

Welcome to the lecture on "Introduction to CSS".

Here are the learning objectives. By the end of this lecture, you should understand the 3 ways of declaring CSS and the basics of how to write CSS rules. CSS is a fundamental tool in web development, as it controls the presentation and layout of web pages.

CSS3 is the latest evolution of Cascading Style Sheets. It brings many new features and enhancements to the way we style web pages. This slide includes a taxonomy of CSS3 that shows the different features that comes with different version of CSS.

As mentioned earlier, HTML defines the content of a web page, while CSS defines the presentation of that content. When we talk about UI/UX design, CSS is a key technology that enhances the user interface and user experience by allowing us to style and layout HTML elements effectively.

You might be thinking, isn't styling just about changing font sizes, colors, and other simple properties? How much of a difference can it really make?

Well, let's take a look at the following example from the CSS Zen Garden to demonstrate how CSS can completely transform the look and feel of a website. The following screenshot highlights 6 distinctively different sites but do you know that they all originated from the same HTML with the exception of just the CSS definition.

There are 3 ways to declare CSS:

First is using the internal stylesheet approach, enclosed between the style tag.

Second is using external stylesheets where the CSS rules are in an external file, such as styles.css.

Third is using inline styles by defining the style within the STYLE attribute of an HTML element.

In the first 2 approaches, we create a list of CSS rules that apply styles across the HTML document.

A CSS rule consists of a selector and a series of property-value pairs, each terminated with a semicolon.

For example, body could be a selector, and background-color and font-size are properties with their respective values.

It is also possible to apply the same styling rules to multiple selectors by separating the selectors with a comma.

Additionally, you can add comments to your CSS using the following syntax. Just remember that the double slash syntax commonly used in many programming languages is not a valid way to add comments in CSS.

Internal stylesheets are typically placed within the head section of your web page. Here is an example of how to define internal stylesheets.

An external stylesheet is a separate CSS file linked to your web page. Essentially, they are the same as how we would define for internal stylesheet except that rather than putting the styling information within the HTML document, we put the CSS rules in a separate file with a css extension. We then use the link tag to associate this webpage with the external stylesheet, making the final effect almost equivalent to the internal stylesheet approach.

The third approach is to use inline styles where we apply the styling directly to specific HTML elements using the STYLE attribute.

An example is shown here. Unlike the previous 2 approaches, we do not specify any selectors but only put the list of CSS properties and values. The reason why we do not need to specify any selectors is because the style will be apply to this HTML element containing the STYLE attribute.

Inline styles are useful for applying unique styles to a single element, but they are less efficient for larger projects where consistency and maintainability are key.

To summarize, we have covered the 3 ways of declaring CSS: internal stylesheets, external stylesheets, and inline styles. We also learned the basics of writing CSS rules, including selectors and property-value pairs.

Video 7: Introduction to Document Object Model (DOM)

Welcome to the lecture on "Introduction to Document Object Model ". Here are the learning objectives. By the end of this lecture, you should understand what the Document Object Model or DOM is, and appreciate how to make use of the browser developer tool to inspect and modify the DOM.

The Document Object Model or DOM, is a programming interface for web pages. When a web page is loaded, the browser creates a DOM of the page. This DOM allows developers to programmatically access and update the content, structure, and style of a web page. Essentially, the DOM is a representation of the document as a structured group of nodes and objects.

The HTML DOM is a tree of HTML elements, often called the DOM tree. Imagine it like a family tree where the HTML elements are the nodes connected by branches. This tree structure allows us to navigate through the document and manipulate its elements easily.

To access the DOM tree, you can use the browser's developer tool. On Firefox or Chrome, press F12 to open these tools. This tool allows you to view and edit the DOM. It’s a powerful feature that can help you understand how your web page is structured and how it can be modified dynamically.

Technically speaking, what the browser developer tool shows you is the DOM, not the page source. When the browser loads a page, it creates a DOM from the HTML and uses this DOM to render the page you see. This means any changes you make in the developer tool are applied directly to the DOM, and you can see the results immediately.

Using the inspect element feature in the browser developer tool, you can select any element in the DOM tree. This is incredibly useful for seeing how different elements are styled and structured. By inspecting an element, you can view its HTML and CSS, making it easier to debug and understand how a particular element is rendered.

Once you've inspected an element, you can also modify the DOM on the fly. This means you can change both the HTML and CSS to see how the page would look after modifications. It is important to note that these changes do not modify the actual files on the server. They are temporary and only exist in your current session of your browser.

To summarize, we have covered what the Document Object Model is. We have also explored how to use the browser developer tool to inspect and modify the DOM, allowing us to see changes in real-time without affecting the actual source files.

Video 8: Basic CSS Selectors

Welcome to the lecture on "Basic CSS Selectors". Here are the learning objectives. By the end of this lecture, you should understand the different types of basic CSS selectors. These selectors are fundamental in applying styles to specific elements on your web page.

First, let's talk about element selectors. Element selectors select all elements of a given HTML tag. For example, if we want to apply a style to all H1 elements, we can use an element selector like this.

Next, we have class selectors. Class selectors select all elements with a specific CLASS attribute. For instance, if we want to make the text of an element red, we can apply this red color style to an element by specifying red for the CLASS attribute of that element.

Note it is possible to have more than one class value by separating the different class values using a space.

Next, let's move on to ID selectors. ID selectors select a single element with a specific ID attribute. Here's an example. ID selectors start with the hex sign.

In our HTML, it works similarly to the class selectors except that now we are considering the ID attribute. However, unlike class selectors, note that the ID value should be unique so this title ID selector should only match at most one element in the page.

Sometimes we might want to match all elements in the page. We could use the universal selectors. The asterisk here can be thought of as matching any HTML tag. This rule will make every element on the page red. While powerful, universal selectors should be used sparingly to avoid unintended consequences.

Selectors can also be combined. For example, to style all DIV elements with the red class, you can write it as the following.

You can also apply the same style to multiple selectors using a comma. This rule will apply the red color to both elements with the IDs id1 and id2.

When it comes to resolving CSS properties, the style of a parent element is often inherited by its child elements. For example, given the following CSS rules.

And the following HTML

This will result in the parent text having a grey background, with child1 text being red.

Adding more styles to the previous example…

Here, the parent has a grey background, child1 has red text, and child2 has light blue text with a blue background.

Inheritance is a key concept in CSS. Some properties are inherited by default by the child elements, while others are not. The complete specification for inheritance is detailed on the W3C website. You can refer to the W3C website to check whether a property is inherited.

To summarize, we have covered the different types of basic CSS selectors: element selectors, class selectors, ID selectors, and universal selectors. We have also discussed how to combine selectors and resolve CSS properties.

Understanding these selectors is fundamental for styling your web pages effectively.

Video 9: Common CSS Properties

Welcome to the lecture on "Common CSS Properties". Here are the learning objectives. By the end of this lecture, you should understand what CSS properties are and how to use some of the most common ones. This knowledge is crucial for effectively styling HTML elements on your web pages.

First, let's recall what CSS properties are. CSS properties are used to style HTML elements. Each property consists of a name and a value, and they are applied using a selector.

For example, you can set the color of all paragraphs to blue like this.

Let's start with text-related properties. One of the most commonly used properties is text-align. This property sets the horizontal alignment of text within an element. Here's how you can align text to the left. Other possible values are RIGHT, CENTER, and JUSTIFY.

Next, we have text-decoration. This property is used to set the appearance of decorative lines on the text. For example, you can underline text like this. Other values include NONE, OVERLINE, and LINE-THROUGH.

Another useful property is text-transform. This property controls the capitalization of text. To convert text to uppercase, you can use the UPPERCASE value. Other values are LOWERCASE and CAPITALIZE.

Text indentation is controlled by the text-indent property. This property specifies the indentation of the first line of text within an element. For example, this will indent the first line of text by 100 pixels.

Text spacing can be adjusted using several properties.

letter-spacing adjusts the space between characters,

word-spacing adjusts the space between words,

and line-height adjusts the space between lines of text.

Color-related properties are essential for styling. The COLOR property sets the text color, and the background-color property sets the background color. Here are some examples how to set the color and background color.

There are multiple ways to define color values in CSS. We can define color values using its color name like red or blue. We can also define the color value as RGB values using the RGB function and specifying the red, green, and blue values respectively. Each of these values ranges from from 0 to 255. Another way to define the RGB value is to use an hexadecimal values starting with # followed by three pairs of hexadecimal digits as shown here. In this last example, the red component of the background color is set to hexadecimal of FF which is 255 and hexadecimal of 66 for the green component which is 102 in decimal form and 0 for the blue component.

The background property allows for shorthand notations for setting multiple background properties at once. For example, we could set the background-color, background-image, background-repeat, background-attachment, and background-position at the same time.

To summarize, we have covered what CSS properties are and how to use some of the most common ones, including text-related properties, color-related properties and background properties. Understanding these properties is fundamental to effectively styling your web pages.

Video 10: Module Summary

Welcome to the module summary for "HTML and CSS Fundamentals". In this lecture, we will recap the key points covered for each topic in this module. We began with discussing about the structure and syntax of HTML pages. By now you should know that HTML stands for HyperText Markup Language, used to create web pages and it is a scripting language used to create web pages. A web page typically consist of the head and body section. The head section contains meta information and title of the page while the body section include content visible to the users. Remember that tags usually come in pairs and attributes provide additional information about the tag.

Next, for the commonly used HTML tags, we discussed the text related tag such as headings and paragraphs, hyperlinks, image, lists, tables, divs and spans.

URLs are crucial for linking resources. They can be absolute, including the full path, or relative, referring to a location relative to the current document. Use absolute URLs for external links and relative URLs for internal links.

Forms are essential for user interaction. The FORM tag encloses input elements, and the ACTION attribute specifies where the form data goes. The METHOD attribute defines how data is sent, using GET or POST. Input types include TEXT, PASSWORD, FILE, EMAIL, and more. Placeholders, and validation attributes enhance usability.

We then went on to discuss about how to apply styling to web pages using CSS. There are 3 ways to apply CSS: inline styles, internal stylesheets, and external stylesheets. CSS rules consist of selectors and declarations. The selector is used to target HTML elements, while declarations style them with properties and values.

The DOM represents the structure of a webpage as a tree of objects. It allows you to access and manipulate HTML elements programmatically.

CSS selectors target HTML elements to apply styles. We have covered element selectors, class selectors, ID selectors, and universal selectors.

We have also covered common CSS properties include text-align, text-decoration, color, and background-color. Try experimenting with these values to achieve the desired look.

In summary, we've covered HTML structure and syntax, common tags, URLs, forms, an introduction to CSS and the DOM, basic selectors, and common CSS properties. These fundamentals are essential for building and styling web pages effectively. Thank you for your attention. We hope that this module has provided you with a good foundation into writing web pages.