Semantic HTML

Task

Visit our website

Introduction

In this task, you'll dive into semantic HTML and learn why it's an essential part of modern web development. We'll cover a list of common semantic elements and explore how they enhance accessibility, improve search engine optimisation, and make your code easier to understand. By mastering semantic HTML, you'll be able to create more meaningful and well-structured web pages.

Semantics

One of the most important features of HTML5 is its semantics. The word **semantic** means “relating to meaning”. HTML was originally designed to semantically describe scientific documents, and it has since evolved to describe much more. Semantic HTML refers to writing HTML in a way that is more comprehensible by better defining the different sections and layout of web pages.

When using semantic HTML, we choose HTML elements based on their meaning, not on how they are presented. It also makes web pages more informative and adaptable, allowing browsers and search engines to better interpret content. Elements such as `<div>` and `<span>` are not semantic elements since they provide no context as to what is inside of those tags.

List of common semantic HTML elements

There are so many semantic HTML elements that can be used to create clear, meaningful web content. These elements help organise a web page, making it easier for search engines and users to understand its structure. For a deeper dive into the concept of semantics in HTML, check out this guide on **MDN Web Docs**.

Below, you’ll find some of the most common and useful semantic HTML elements along with examples to illustrate how they work.

`<article>` – This defines independent and self-contained content in a web page that is intended to be reusable.

Example use cases:

- A forum post
- A magazine or newspaper article
- A blog entry
- A product card
- A user-submitted comment

Here is a demonstration of how to use the `<article>` element:

```
<article class="forecast">
    <h1>Weather forecast for Seattle</h1>
    <article class="day-forecast">
        <h2>03 March 2022</h2>
        <p>Rain.</p>
    </article>
    <article class="day-forecast">
        <h2>04 March 2022</h2>
        <p>Periods of rain.</p>
    </article>
    <article class="day-forecast">
        <h2>05 March 2022</h2>
        <p>Heavy rain.</p>
    </article>
</article>
```

`<aside>` – This defines a portion of a web page that has content that is only indirectly related to the web page's main content.

The following example uses `<aside>` to mark up a paragraph in an article that is only indirectly related to the main article content.

`<cite>` – This defines the title of a creative work. An example of this can also be seen below:

```
<article>
  <p>
    The Disney movie titled <cite>The Little Mermaid</cite> was
    first released in theatres in 1989.
  </p>
  <aside>
    <p>
      The movie earned $87 million during its initial release.
    </p>
  </aside>
  <p>
    More info about the movie...
  </p>
</article>
```

`<details>` – This creates a widget in which information is visible only when the widget is toggled into an "open" state. A summary or label must be provided using the `<summary>` element, described further below.

Here is an example of the `<details>` element:

```
<details>
    <summary>Details</summary>
    Something small enough to escape casual notice.
</details>
```

`<summary>` – This specifies a summary, caption, or legend for a `<details>` element's disclosure box. Clicking on the `<summary>` element toggles the state of the parent `<details>` element between open and closed. See the example above under the `<details>` heading for an example of the `<summary>` element.

`<figure>` – This represents self-contained content like illustrations, diagrams, photos, and code listings that are specified using the `<figcaption>` element.

`<figcaption>` – This represents a caption or legend describing the rest of the contents of its parent `<figure>` element.

Here is an example of the `<figure>` and `<figcaption>` element:

```
<figure>
    <img
src="https://hips.hearstapps.com/hmg-prod/images/womanyellingcat-1573233850.jp
g"
         alt="A cat meme template">
    <figcaption>A cat meme template</figcaption>
</figure>
```

`<footer>` – This defines a footer for a web page or section. A `<footer>` typically contains information about the author of the section, copyright data, or links to related web pages.

Here is an example of the `<footer>` tag:

```
<article>
    <h1>How to be a wizard</h1>
    <ol>
        <li>Grow a long, majestic beard.</li>
        <li>Wear a tall, pointed hat.</li>
        <li>Have I mentioned the beard?</li>
    </ol>
    <footer>
        <p>(c) 2018 Gandalf</p>
    </footer>
</article>
```

`<header>` – The `<header>` element represents introductory content, typically a group of introductory or navigational aids.

The `<header>` element may contain:

- Heading elements
- A logo
- A search form
- An author name

Here is an example of the `<header>` element:

```
<header class="page-header">
    <h1>Cute Cat Express!</h1>
</header>

<main>
    <p>I love cats <em>so</em> much! Like, really, a lot. They're adorable and
so, so snuggly soft!</p>
</main>
```

`<main>` – The `<main>` element represents the content of the `<body>` of a web page. The main content area consists of content that is directly related to or expands upon the central topic of a web page, or the central functionality of an application. Please look at the example under the `<header>` element for an example of the `<main>` element.

`<nav>` – The `<nav>` element represents a section of a web page that provides navigation links to either within the current web page or to other web pages.

Example use cases of navigation sections:

- Menus
- Tables of contents
- Indexes

Here is an example of the `<nav>` element:

```
<nav>
    <a href="/">Home</a>
    <a href="#about">About us</a>
    <a href="#articles">Articles</a>
    <a href="#contact">Contacts</a>
</nav>
```

`<section>` – This represents a generic standalone section of a web page that doesn't have a more specific semantic element to represent it. Sections should always have a heading, with very few exceptions.

Here is an example of the `<section>` element:

```
<section>
  <h1>Pink Flamingo</h1>
  <p>Plastic flamingos have been used as garden ornaments since the late
1950s.</p>
</section>
```

Why use semantic HTML?

Semantic HTML is more than a coding standard; it's a practice that provides clear benefits to developers, users, and search engines. Let's explore how using semantic HTML can elevate your web projects.

Accessibility

Semantic HTML is essential for making web content accessible to all users, especially those with disabilities. Elements such as `<header>`, `<main>`, `<section>`, and `<nav>` provide a meaningful structure that assistive technologies can interpret, allowing users to navigate a page efficiently.

For example, using the `<nav>` element signals to screen readers that the section contains navigation links. This allows users relying on screen readers to jump directly to the navigation section if they want, improving the browsing experience.

Examine the following code snippet:

```
<header>
  <h1>Welcome to Our Personal Development Blog</h1>
</header>
<nav aria-label="Main Navigation">
  <ul>
      <li><a href="#growth-strategies">Growth Strategies</a></li>
      <li><a href="#wellness-tips">Wellness Tips</a></li>
      <li><a href="#contact">Contact Us</a></li>
  </ul>
</nav>
<main>
  <section id="growth-strategies">
      <h2>Top Personal Growth Strategies for 2024</h2>
      <p>Discover effective ways to reach your full potential this year...</p>
  </section>
</main>
<footer>
  <p>&copy; 2024 Personal Development Blog. All rights reserved.</p>
</footer>
```

When building a website, consider how someone using only a keyboard or a screen reader would interact with your content. Would they be able to skip to the main content quickly? Semantic tags like `<nav>` and `<main>` ensure they can, fostering an inclusive experience.

Search engine optimisation (SEO)

Originally, HTML was designed to define the structure of web pages without much focus on meaning. This made it difficult for search engines to differentiate between main content and less important elements.

When semantic tags are used, search engines don't just see raw text and basic layout. They see labelled sections that clarify what each part of the content represents, like articles, headings, or navigation menus. This better understanding allows them to index the content more effectively and rank it according to its relevance for users' search queries.

Semantic HTML improves SEO by helping search engines better understand and index your content. When you use tags like `<article>`, `<section>`, and `<header>`, search engines can identify the structure of your content and prioritise it accordingly.

For example, suppose you're writing a blog post. Wrapping your post in an `<article>` element helps search engines understand that the content is a standalone piece of information, which can be beneficial for search rankings.

Take a look at the following code example:

```
<article>
  <header>
      <h2>5 Key Tips for Personal Development</h2>
      <p>By John Doe | November 12, 2024</p>
  </header>
  <section>
      <h3>1. Set Clear Goals</h3>
      <p>Define your objectives to create a roadmap for personal growth...</p>
  </section>
  <section>
      <h3>2. Develop a Routine</h3>
      <p>Consistent habits help reinforce positive changes and build
momentum...</p>
  </section>
  <footer>
      <p>Share this article to inspire others!</p>
  </footer>
</article>
```

Think about how search engines rank pages. Using semantic elements helps Google identify key parts of your article, like the title in the `<header>` and each section in `<section>`, making it more likely that your page appears in relevant searches or featured snippets.

Improved code readability and maintenance

Semantic HTML makes code more readable and maintainable by using descriptive tags. This means developers can quickly understand the layout and purpose of the content.

For example, using `<section>` instead of a generic `<div>` can signal to other developers that the block contains a distinct part of the content.

Here's a code example to consider:

```
<section>
  <h2>Customer Testimonials</h2>
  <p>Here's what our clients have to say:</p>
  <blockquote>
      <p>"This product changed the way I work--absolutely incredible!"</p>
      <footer>-- Alex Johnson, Product Designer</footer>
  </blockquote>
</section>
```

Imagine joining a new project where the previous developer used only `<div>` tags. It would take longer to decipher the content structure compared to semantic tags that outline the document's meaning. Clear use of semantic tags reduces onboarding time and the likelihood of errors.

Practical task

Create a simple web page showcasing your favourite hobby using semantic HTML. Your page should include:

- At least these semantic HTML elements: `<footer>`, `<header>`, `<main>`, `<nav>`, and `<section>`.
- At least three other HTML5 tags like: `<article>`, `<aside>`, `<figure>`, etc.

Save the HTML file in a folder named after your hobby (e.g., **hobby/** with **index.html** inside).

Important: Be sure to upload all files required for the task submission inside your task folder and then click "Request review" on your dashboard.

Share your thoughts

Please take some time to complete this short feedback **form** to help us ensure we provide you with the best possible learning experience.
