

HTML Fundamentals: Building Your First Website

An Interactive Learning Journey

Target Audience: Undergraduate Students
Learning Format: Conversational Workbook

Table of Contents

1. Real-Time Scenario: Building a Personal Portfolio Website
 2. Conversational Learning Journey
 3. Core Concepts Deep Dive
 4. Hands-On Exercises
 5. Step-by-Step Solutions
 6. Summary and Next Steps
-

Real-Time Scenario - Building a Personal Portfolio Website

Meet Your Learning Partners

User: Alex, a computer science undergraduate student who needs to create a personal portfolio website for internship applications.

Expert: Dr. Sarah Chen, a web development instructor with 10 years of experience in teaching HTML and web technologies.

The Challenge

Alex has just received an email from the career services office at their university. The message is clear: "All students applying for tech internships this summer must submit a personal portfolio website showcasing their projects and skills." Alex has never built a website before and feels overwhelmed by the task ahead.

Alex decides to reach out to Dr. Chen, who has offered to mentor students through their first web development projects. The conversation begins in Dr. Chen's office on a Tuesday afternoon, with Alex's laptop open and ready to start coding.

The Learning Conversation Begins

Alex: Dr. Chen, I'm really nervous about this portfolio website requirement. I've heard about HTML, but I don't even know where to start. What exactly is HTML, and how does it work?

Expert: That's a great question, Alex! Let me ease your concerns first - HTML is actually quite logical and beginner-friendly. **HTML stands for HyperText Markup Language.** Think of it as the skeleton or foundation of every website you've ever visited. Just like how a **house needs a frame** before you can add walls, windows, and decorations, a **website needs HTML structure** before you can add styling and interactive features.

Alex: Okay, that makes sense as an analogy. But what does "markup language" actually mean? Is it like a programming language?

Expert: Excellent question! HTML is not exactly a programming language like Python or Java. Instead, it's a **markup language**, which means it's **used to annotate and structure content**. Think of it like editing a document with a red pen - you're marking up the text to indicate what each part should be. In HTML, we use special codes called "tags" to tell the browser "this is a heading," "this is a paragraph," "this is a link," and so on.

Alex: I see. So **HTML tags are like instructions for the browser?**

Expert: Exactly! The browser reads these instructions and displays the content accordingly. For example, when you write `<h1>Welcome to My Portfolio</h1>`,

you're telling the browser "display this text as a main heading." The browser then makes it large and bold automatically.

Alex: That's starting to make sense. But how do I actually create an HTML file? Do I need special software?

Expert: Here's the beautiful thing about HTML - you can create it with any simple text editor! You could use Notepad on Windows, TextEdit on Mac, or more advanced editors like Visual Studio Code. The key is to save your file with a `.html` extension, like `index.html` or `portfolio.html`.

Alex: Wait, why `index.html` specifically?

Expert: Great observation! `index.html` is a special filename that web servers recognize as the default page for a website. When someone visits your website's main address, the server automatically looks for and displays the `index.html` file. It's like the front door of your digital house.

Alex: This is really helpful! But before we start coding, can you show me what a basic HTML document looks like? I want to understand the overall structure first.

Expert: Absolutely! Understanding the structure is crucial. Let me show you the anatomy of an HTML document, and then we'll build your portfolio step by step.

Core Concepts Deep Dive

The Basic HTML Document Structure

Expert: Every HTML document follows a fundamental structure. It's like the blueprint for your webpage. Let's break it down:

```

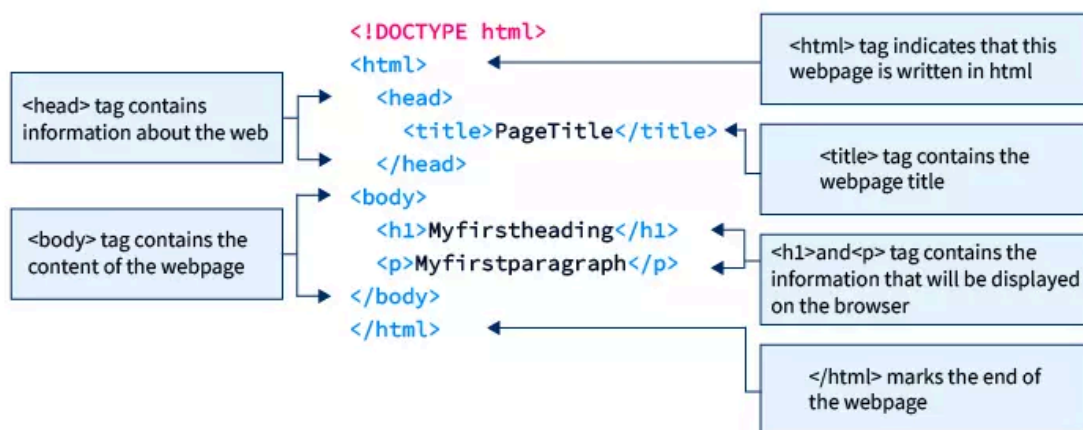
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>My Awesome Portfolio</title>
</head>
<body>
  <!-- Content goes here -->
  <h1>Welcome to My Portfolio!</h1>
  <p>This is where I'll showcase my projects.</p>
</body>
</html>

```

Expert: Let's go through each part:

- **<!DOCTYPE html>**: This declaration defines the document type and helps the browser display web pages correctly. It tells the browser that this is an HTML5 document.
- **<html lang="en">**: This is the root element of an HTML page. All other elements are contained within it. The `lang="en"` attribute specifies the language of the document, which is good for accessibility and search engine optimization.
- **<head>**: This section contains meta-information about the HTML document, such as its title, character set, styles, and scripts. This content is *not* displayed on the web page itself but is crucial for how the browser handles the page.
 - **<meta charset="UTF-8">**: Specifies the character encoding for the document, ensuring proper display of various characters.
 - **<meta name="viewport" content="width=device-width, initial-scale=1.0">**: This is vital for responsive web design, telling the browser how to control the page's dimensions and scaling on different devices.
 - **<title>**: The text inside this tag appears in the browser's title bar or tab. It's also what search engines use as the title for your page in search results.
- **<body>**: This is where all the visible content of your web page resides. Everything you see on a website – text, images, links, videos – is placed within the `<body>` tags.

Expert: Take a look at this diagram to visualize the structure:



SCALER
Topics

Alex: So, the `<head>` is like the behind-the-scenes setup, and the `<body>` is the actual show?

Expert: Precisely! You've got the analogy down. Now, let's talk about the building blocks within the `<body>` – HTML elements and tags.

HTML Elements and Tags

Expert: An **HTML element** is defined by a start tag, some content, and an end tag. For example, `<p>This is a paragraph.</p>` is a paragraph element. **Some elements are self-closing**, meaning they don't have a separate closing tag, like `` for images or `
` for a line break.

Alex: So, a tag is just the name inside the angle brackets, like `p` or `img`?

Expert: Yes, that's the tag name. The combination of the opening tag, content, and closing tag forms an element. And those extra bits inside the opening tag, like `src` and `alt` in the `` example, are called **attributes**.

Alex: What are attributes for?

Expert: Attributes provide additional information about an element. They modify the default behavior or appearance of an element. For instance, the `src` attribute in an `` tag specifies the source (URL) of the image, and the `alt` attribute provides alternative text for the image, which is important for accessibility if the image can't be displayed.

Here's a visual guide to some common HTML tags:

Heading Tags	<code><h1></code> , <code><h2></code> , <code><h3></code> , <code><h4></code> , <code><h5></code> , and <code><h6></code>
Paragraph Tag	<code><p></code>
Line Break Tag	<code>
</code> or <code>
</code>
Center Tag	<code><center></code>
Horizontal Rule Tag	<code><hr></code> or <code><hr /></code>
Preserve Formatting Tag	<code><pre></code>
Non-breaking Space	<code>&nbsp;</code> ; <code>&lt;</code> → <code><</code> (less than sign) <code>&</code> for Entity
Listing Tags	<code></code> and <code></code> <ul style="list-style-type: none">• Unordered Listing: <code></code>• Ordered Listing: <code></code>• List Item: <code></code>
HTML Basic Tags	

Semantic HTML

Expert: As you progress in web development, you'll hear a lot about 'semantic HTML.' This means using HTML tags that accurately describe the purpose or meaning of the content they enclose, rather than just how they look.

Alex: Why is that important? If it looks the same, does it matter what tag I use?

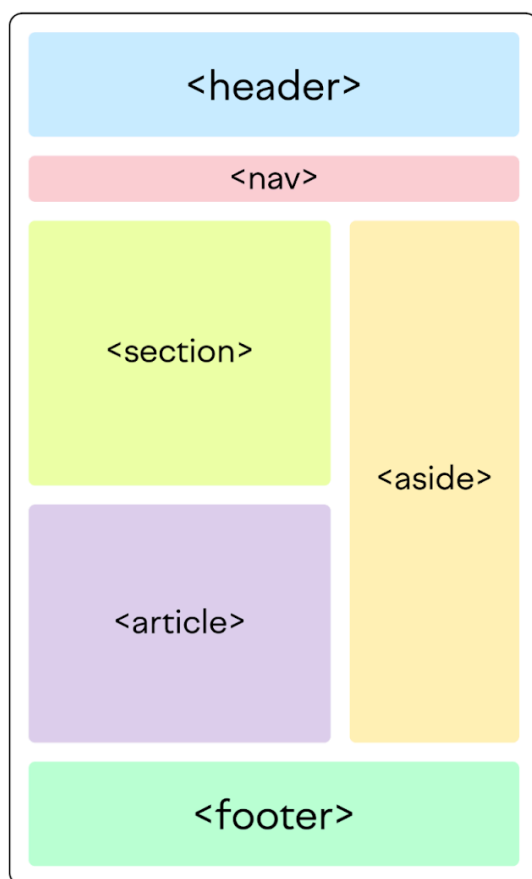
Expert: It matters a lot! Semantic HTML improves accessibility for users with disabilities (screen readers rely on it), helps search engines understand your content better (which can improve your website's ranking), and makes your code easier for other developers (and your future self!) to understand and maintain.

For example, instead of using a generic `<div>` tag for everything, you'd use `<header>` for your website's header, `<nav>` for navigation links, `<article>` for independent content, `<section>` for thematic groupings, and `<footer>` for the footer.

Here's a comparison of non-semantic vs. semantic HTML structure:



Semantic HTML



Expert: Using semantic tags helps create a clear outline of your document, much like a well-structured essay has an introduction, body paragraphs, and a conclusion. It gives meaning to your content beyond just its visual presentation.

Alex: That makes a lot of sense. So, it's about making the code more meaningful, not just functional.

Expert: Exactly! Now that you have a grasp of the basic structure and the importance of semantic elements, are you ready to try building a simple HTML page for your portfolio?

Alex: Definitely! I'm excited to put this into practice.

: Hands-On Exercises

Exercise 1: Your First HTML Page (Easy)

Your goal is to create a basic `index.html` file for Alex's portfolio. This page should:

1. Include the basic HTML5 document structure (`<!DOCTYPE html>` , `<html>` , `<head>` , `<body>`).
2. Set the page title to "Alex's Portfolio".
3. Inside the `<body>` , add a main heading (`<h1>`) that says "Welcome to My Portfolio".
4. Below the heading, add a paragraph (`<p>`) that says "This is a simple page to showcase my projects and skills."

Exercise 2: Adding Semantic Structure (Medium)

Now, let's enhance Alex's `index.html` by adding some semantic HTML elements. Modify the previous `index.html` file to include:

1. A `<header>` section containing the `<h1>` heading.
2. A `<nav>` section immediately after the `<header>` , which will eventually hold navigation links (for now, just add a comment inside it: `<!-- Navigation links will go here -->`).
3. A `<main>` section that wraps the `<p>` paragraph.
4. A `<footer>` section at the bottom of the `<body>` , containing a copyright notice (e.g., `<p>© 2025 Alex. All rights reserved.</p>`).

Step-by-Step Solutions Dialogue

Solution for Exercise 1: Your First HTML Page

Expert: Alright, Alex, let's tackle the first exercise. You need to create a new file named `index.html` and set up the basic structure. What's the very first line you should add to declare it as an HTML5 document?

Alex: That would be `<!DOCTYPE html>`, right?

Expert: Perfect! And what comes next, wrapping all the content?

Alex: The `<html>` tags, with `lang="en"` for English.

Expert: Excellent. Inside the `<html>` tags, we have two main sections. Which one comes first, and what's its purpose?

Alex: The `<head>` section, for metadata like the character set, viewport settings, and the page title.

Expert: Spot on! And what about the title itself? What tag do you use for that, and what should the title be?

Alex: `<title>Alex's Portfolio</title>`.

Expert: You're doing great! Now, for the visible content of the page, which section do we use?

Alex: The `<body>` section.

Expert: Correct. And inside the `<body>`, the exercise asks for a main heading and a paragraph. How would you write the main heading?

Alex: `<h1>Welcome to My Portfolio</h1>`.

Expert: Exactly. And the paragraph?

Alex: `<p>This is a simple page to showcase my projects and skills.</p>`.

Expert: Fantastic! Let's put it all together. Here's what your `index.html` should look like for Exercise 1:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Alex's Portfolio</title>
</head>
<body>
  <h1>Welcome to My Portfolio</h1>
  <p>This is a simple page to showcase my projects and skills.</p>
</body>
</html>
```

Expert: How does that feel? Did you get something similar?

Alex: Yes, that's almost exactly what I had! It's cool to see it all come together.

Solution for Exercise 2: Adding Semantic Structure

Expert: Now for Exercise 2, we're going to make that page more meaningful using semantic HTML. The first step is to wrap your `<h1>` in a `<header>` tag. What do you think that looks like?

Alex: So, it would be `<header><h1>Welcome to My Portfolio</h1></header>`?

Expert: Precisely. Next, we need a navigation section. What semantic tag should we use for that, and what should go inside it for now?

Alex: The `<nav>` tag, and inside it, a comment: `<!-- Navigation links will go here -->`.

Expert: Excellent. Now, for the main content of the page, which is your paragraph, what semantic tag should wrap it?

Alex: That would be `<main>`.

Expert: Correct. And finally, for the copyright notice at the bottom, what semantic tag is appropriate, and how would you write the copyright paragraph?

Alex: The `<footer>` tag, and inside it, `<p>© 2025 Alex. All rights reserved.</p>`.

Expert: You've got it! Let's see the complete code for Exercise 2:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Alex's Portfolio</title>
</head>
<body>
  <header>
    <h1>Welcome to My Portfolio</h1>
  </header>
  <nav>
    <!-- Navigation links will go here -->
  </nav>
  <main>
    <p>This is a simple page to showcase my projects and skills.</p>
  </main>
  <footer>
    <p>&copy; 2025 Alex. All rights reserved.</p>
  </footer>
</body>
</html>
```

Expert: How was that? Do you see how using these tags makes the structure of the document clearer and more organized?

Alex: Yes, it really does! It feels much more professional and easier to understand at a glance. This is great, Dr. Chen!

Summary and Next Steps

Expert: Today, Alex, you've taken your first significant steps into the world of web development. You've learned:

- What HTML is and its role as the foundation of web pages.
- The basic anatomy of an HTML document, including `<!DOCTYPE html>`, `<html>`, `<head>`, and `<body>`.
- The concept of HTML elements, tags, and attributes.
- The importance and application of semantic HTML for better structure, accessibility, and SEO.
- How to create a basic HTML page and incorporate semantic elements through hands-on exercises.

Alex: I feel much more confident now! Thank you, Dr. Chen.

Expert: You're very welcome, Alex. For your next steps, I recommend:

1. Experimenting with more HTML tags (e.g., `<a>` for links, `` for images, `` and `` for lists).
2. Exploring online resources like MDN Web Docs or W3Schools for a comprehensive HTML tag reference.
3. Thinking about the content you want to include in your actual portfolio and how you would structure it using the semantic tags we discussed.

HTML Tables: Organizing Tabular Data



Name	Age	Country
Harry Depp	28	Britain
John Smith	35	USA
Ram Krishna	19	Nepal

Expert: Alex, as your portfolio grows, you might need to display data in a structured, tabular format. This is where HTML tables come in. They are perfect for presenting data in rows and columns.

User: Like a spreadsheet?

Expert: Exactly! Tables are defined with the `<table>` tag. Inside, you use `<tr>` for table rows, `<th>` for table headers, and `<td>` for table data (cells).

Here's a basic structure:

```
<table>
  <thead>
    <tr>
      <th>Name</th>
      <th>Age</th>
      <th>City</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Alice</td>
      <td>30</td>
      <td>New York</td>
    </tr>
    <tr>
      <td>Bob</td>
      <td>24</td>
      <td>London</td>
    </tr>
  </tbody>
</table>
```



Create Tables in HTML

Column 1	Column 2	Column 3
Row 1 Cell 1	Row 1 Cell 2	Row 1 Cell 3
	Row 2 Cell 2	Row 2 Cell 3
Row 3 Cell 1		

Expert: The `<thead>` and `<tbody>` tags are optional but highly recommended for semantic structure, especially for accessibility and styling. `<th>` elements are typically bold and centered by default, indicating they are headers for their respective columns or rows.

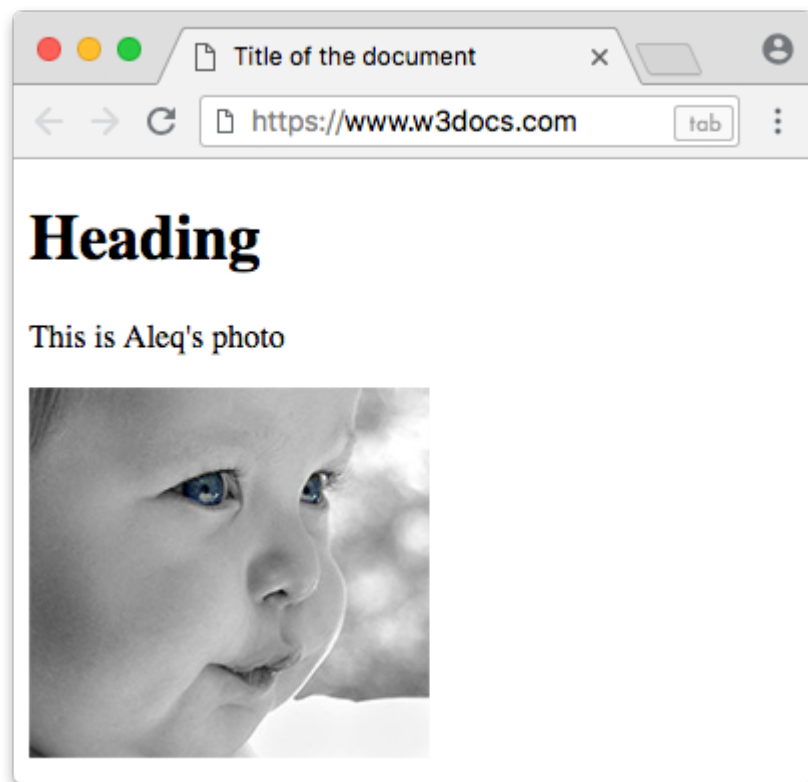
User: Can I make cells span multiple rows or columns?

Expert: Yes, you can! The `colspan` attribute merges cells horizontally, and `rowspan` merges them vertically. For example:

```
<table>
  <tr>
    <th colspan="2">Name</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>First</td>
    <td>Last</td>
    <td>25</td>
  </tr>
</table>
```

Expert: Tables are powerful for data presentation, but remember, they are *not* for layout. Always use CSS for page layout, and tables only for actual tabular data.

HTML Images: Adding Visuals to Your Page



Expert: A picture is worth a thousand words, and on the web, images are crucial for engaging your audience. The `` tag is used to embed an image in an HTML page.

User: How do I tell it which image to show?

Expert: The `` tag is an empty tag, meaning it has no closing tag. It has two essential attributes:

- `src` : Specifies the path to the image file.
- `alt` : Provides alternative text for the image, which is crucial for accessibility (screen readers) and when the image cannot be displayed.

```

```

Expert: You can also control the `width` and `height` of the image using attributes, but it's generally better to control these with CSS for more flexibility and responsiveness.

User: What if the image is on another website?

Expert: You can use a full URL for the `src` attribute, like `src="https://example.com/images/logo.png"`. However, it's usually best to host your own images to avoid issues if the external site removes or moves the image.

HTML Video: Embedding Multimedia

```
HTML_video_Tag.html
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>HTML video tag</title>
5   </head>
6   <body>
7     <video controls="controls" autoplay="autoplay" width="600" height="360">
8       <source src="HTML-video.mp4" type="video/mp4" />
9     </video>
10  </body>
11 </html>
```

www.cssinhtml.com

Expert: Beyond static images, you can also embed videos directly into your web pages using the `<video>` tag. This is great for showcasing project demos or personal introductions.

User: So, I don't need YouTube for every video?

Expert: That's right! The `<video>` tag allows native video playback. It typically includes a `src` attribute pointing to the video file, and `controls` to add playback controls (play/pause, volume, etc.).

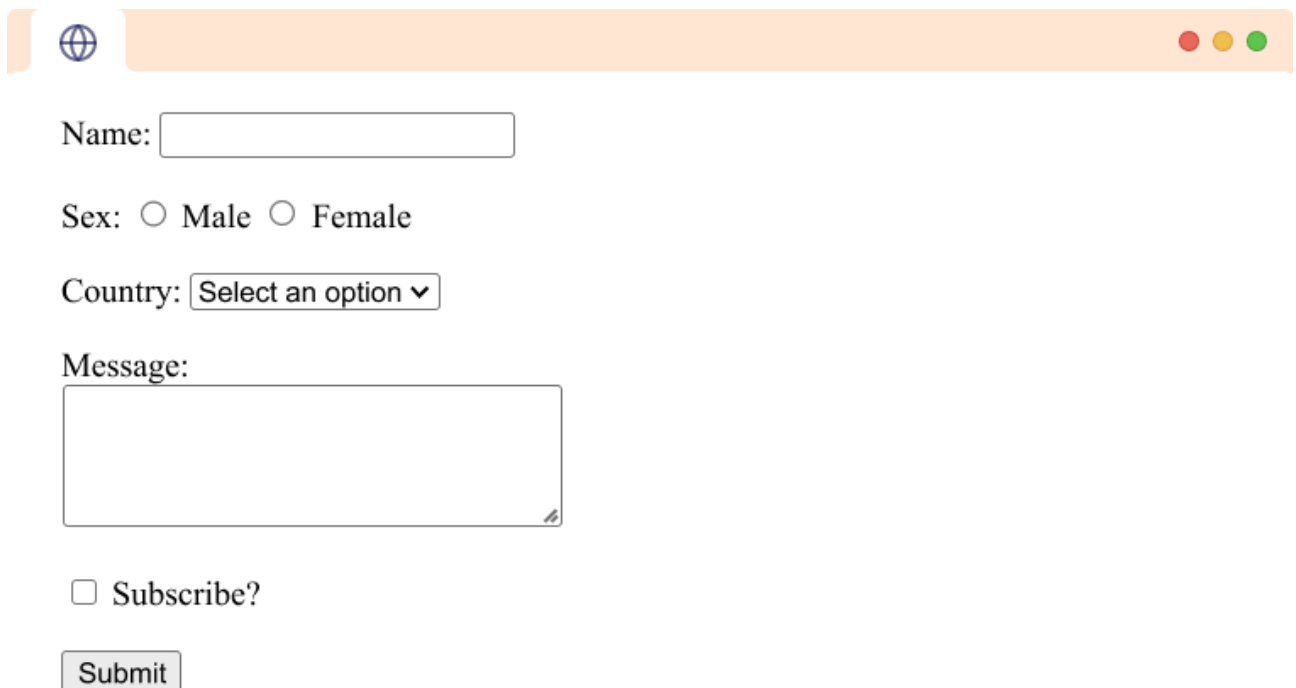
```
<video src="project-demo.mp4" controls width="640" height="360"></video>
```

Expert: For better browser compatibility, you can provide multiple source elements within the `<video>` tag, each pointing to the same video in different formats (e.g., MP4, WebM, Ogg). The browser will use the first format it supports.


```
<video controls width="640" height="360">
  <source src="project-demo.mp4" type="video/mp4">
  <source src="project-demo.webm" type="video/webm">
  Your browser does not support the video tag.
</video>
```

Expert: Other useful attributes include `autoplay` (starts playing automatically), `loop` (plays repeatedly), and `poster` (an image to display before the video starts playing).

HTML Forms: Collecting User Input



Name:

Sex: ☐ Male ☐ Female

Country:

Message:

☐ Subscribe?

Expert: For your portfolio, you might want to include a contact form or a feedback section. `HTML forms are used to collect user input`. The `<form>` tag is the container for all form elements.

User: How does the form send data?

Expert: The `<form>` tag has two important attributes:

- `action`: Specifies where to send the form data when the form is submitted (e.g., a URL to a server-side script).

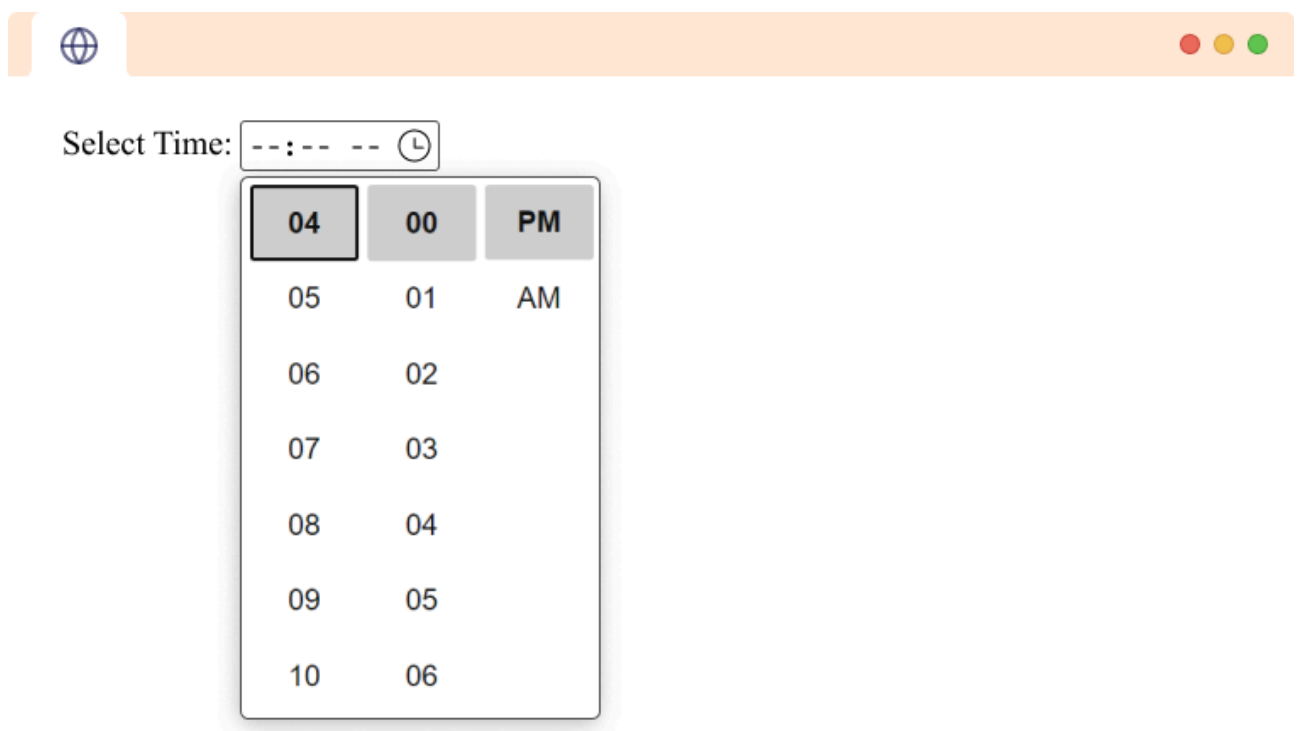
get sends data as url parameters which are visible and ~2000 char limit

- **method** : Specifies the HTTP method to use when sending data (commonly GET or POST). POST is generally preferred for sending sensitive or large amounts of data.

```
<form action="/submit-contact" method="post">
  <!-- Form elements go here -->
</form>
```

Expert: Inside the form, you'll use various input elements to gather different types of data.

HTML Input Tags: Versatile Data Entry



Expert: The `<input>` tag is the most versatile form element. Its behavior changes dramatically based on its `type` attribute.

User: What kinds of inputs can I have?

Expert: Many! Here are some common ones:

- `type="text"` : For single-line text input (e.g., name, email). `html <label for="name">Name:</label> <input type="text" id="name"`

`name="user_name">`

- `type="email"` : For email addresses; browsers might validate the format. `html <label for="email">Email:</label> <input type="email" id="email" name="user_email">`
- `type="password"` : For password input; characters are masked. `html <label for="password">Password:</label> <input type="password" id="password" name="user_password">`
- `type="submit"` : A button to submit the form. `html <input type="submit" value="Send Message">`
- `type="radio"` : For selecting one option from a group. `html <input type="radio" id="html" name="fav_lang" value="HTML"> <label for="html">HTML</label> <input type="radio" id="css" name="fav_lang" value="CSS"> <label for="css">CSS</label>`
- `type="checkbox"` : For selecting zero or more options. `html <input type="checkbox" id="subscribe" name="subscribe" value="yes"> <label for="subscribe">Subscribe to newsletter</label>`

Expert: Always use the `for` attribute in `<label>` and the `id` attribute in the input element to associate them. This is crucial for accessibility.

HTML Select Tags: Dropdown Menus

```
HTML_select_Tag.html x
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>HTML <select> tag</title>
5   </head>
6   <body>
7     <p>Select your favorite Game:</p>
8     <select>
9       <option value="game_1">Game 1</option>
10      <option value="game_2">Game 2</option>
11      <option value="game_3">Game 3</option>
12      <option value="game_4">Game 4</option>
13    </select>
14  </body>
15 </html>
```

www.cssinhtml.com

Expert: When you need to offer a list of predefined options for the user to choose from, the `<select>` tag is your go-to. It creates a dropdown list.

User: How do I add the options to the list?

Expert: Inside the `<select>` tag, you use `<option>` tags for each item in the list. The `value` attribute of the `<option>` is what gets sent to the server, while the text between the tags is what the user sees.

```
<label for="country">Choose your country:</label>
<select id="country" name="user_country">
  <option value="usa">United States</option>
  <option value="can">Canada</option>
  <option value="uk">United Kingdom</option>
  <option value="aus">Australia</option>
</select>
```

Expert: You can also add the `selected` attribute to an `<option>` to make it the default choice, or `disabled` to prevent it from being selected.

HTML Textarea Tags: Multi-line Text Input

```
HTML_textarea_Tag.html
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>HTML textarea tag</title>
5   </head>
6   <body>
7     <form id="myForm">
8       Leave your message: <br>
9       <textarea name="my_textarea" rows="1"></textarea>
10    </form>
11  </body>
12 </html>
```

www.cssinhtml.com

Expert: For longer, multi-line text input, like comments or messages, the `<textarea>` tag is used. Unlike `<input type="text">`, it doesn't have a `value` attribute; its content is placed between the opening and closing tags.

User: Can I control its size?

Expert: Yes, you can! The `rows` and `cols` attributes define the visible height and width of the text area in terms of characters. However, it's often better to control its size and responsiveness using CSS.

```
<label for="message">Your Message:</label>
<textarea id="message" name="user_message" rows="5" cols="30"></textarea>
```

Expert: The text inside the `<textarea>` tags will be the default value displayed to the user. Users can typically resize text areas, but you can prevent this with CSS if needed.

User: This is a lot of new tags! But I can see how they all fit together to build a complete form.

Expert: Exactly, Alex! Each of these elements serves a specific purpose in gathering information from your users. Mastering them will allow you to build interactive and functional web pages. Ready for some hands-on practice?

Hands-On Exercises

Exercise 1: Building a Simple Contact Form (Easy)

Create a new HTML file (e.g., `contact.html`) and build a simple contact form that includes:

1. A text input for **Name**.
2. An email input for **Email**.
3. A textarea for **Message**.
4. A submit button.

Exercise 2: Enhancing the Contact Form (Medium)

Modify the `contact.html` file from Exercise 1 to include:

1. A dropdown (`<select>`) for **Subject** with options like "General Inquiry", "Project Proposal", "Feedback".
2. Radio buttons for **Preferred Contact Method** (e.g., "Email", "Phone").
3. A checkbox for **Subscribe to Newsletter**.

Step-by-Step Solutions Dialogue

Solution for Exercise 1: Building a Simple Contact Form

Expert: Alright, Alex, let's build that simple contact form. We need a text input for the name. How would you start that?

User: I'd use `<label>` and `<input type="text">`.

Expert: Perfect! And for the email?

User: `<label>` and `<input type="email">`.

Expert: Excellent. Now, for the message, which can be multi-line?

User: That would be `<textarea>`.

Expert: Exactly! And finally, how do we add a button to send the form?

User: `<input type="submit">`.

Expert: Fantastic! Let's put it all together. Here's what your `contact.html` should look like for Exercise 1:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Contact Us</title>
</head>
<body>
  <h1>Contact Us</h1>
  <form action="/submit-contact" method="post">
    <label for="name">Name:</label><br>
    <input type="text" id="name" name="user_name"><br><br>

    <label for="email">Email:</label><br>
    <input type="email" id="email" name="user_email"><br><br>

    <label for="message">Message:</label><br>
    <textarea id="message" name="user_message" rows="5" cols="30">
  </textarea><br><br>

    <input type="submit" value="Send Message">
  </form>
</body>
</html>
```

Expert: How does that feel? Did you get something similar?

User: Yes, that's almost exactly what I had! It's cool to see it all come together.

Solution for Exercise 2: Enhancing the Contact Form

Expert: Now for Exercise 2, we're going to enhance this form. First, let's add a dropdown for the subject. What tag would you use for that, and how would you add the options?

User: I'd use `<select>` for the dropdown and `<option>` tags inside it for the choices.

Expert: Perfect! Next, for the preferred contact method, where the user can choose only one option?

User: Radio buttons, using `<input type="radio">` with the same `name` attribute.

Expert: Excellent. And finally, for the newsletter subscription, where the user can opt-in or out?

User: A checkbox, using `<input type="checkbox">`.

Expert: You've got it! Let's see the complete code for Exercise 2:


```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Enhanced Contact Form</title>
</head>
<body>
  <h1>Contact Us</h1>
  <form action="/submit-contact" method="post">
    <label for="name">Name:</label><br>
    <input type="text" id="name" name="user_name" required><br><br>

    <label for="email">Email:</label><br>
    <input type="email" id="email" name="user_email" required><br><br>

    <label for="subject">Subject:</label><br>
    <select id="subject" name="user_subject">
      <option value="general">General Inquiry</option>
      <option value="project">Project Proposal</option>
      <option value="feedback">Feedback</option>
    </select><br><br>

    <label>Preferred Contact Method:</label><br>
    <input type="radio" id="email_contact" name="contact_method"
value="email" checked>
    <label for="email_contact">Email</label><br>
    <input type="radio" id="phone_contact" name="contact_method"
value="phone">
    <label for="phone_contact">Phone</label><br><br>

    <label for="message">Message:</label><br>
    <textarea id="message" name="user_message" rows="5" cols="30" required>
</textarea><br><br>

    <input type="checkbox" id="subscribe" name="subscribe" value="yes">
    <label for="subscribe">Subscribe to Newsletter</label><br><br>

    <input type="submit" value="Send Message">
  </form>
</body>
</html>

```

Expert: How was that? Do you see how adding these elements makes the form more comprehensive and user-friendly?

User: Yes, it really does! This is great, Dr. Chen!

Summary and Next Steps

Expert: Today, Alex, you've taken your first significant steps into the world of web development. You've learned:

- What HTML is and its role as the foundation of web pages.
- The basic anatomy of an HTML document, including `<!DOCTYPE html>`, `<html>`, `<head>`, and `<body>`.
- The concept of HTML elements, tags, and attributes.
- The importance and application of semantic HTML for better structure, accessibility, and SEO.
- How to create a basic HTML page and incorporate semantic elements through hands-on exercises.
- How to use HTML tables to organize tabular data.
- How to embed images and videos into your web pages.
- How to create forms and use various input types, select dropdowns, and text areas to collect user input.

User: I feel much more confident now! Thank you, Dr. Chen.

Expert: You're very welcome, Alex. For your next steps, I recommend:

1. Experimenting with more HTML tags and attributes.
 2. Exploring online resources like MDN Web Docs or W3Schools for a comprehensive HTML tag reference.
 3. Thinking about the content you want to include in your actual portfolio and how you would structure it using the semantic tags we discussed.
-