# **Website Designing (Module – 5 HTML5)**

**(Q.1)** **What are the new tags added in HTML5?**

**(ANS):**

HTML5 introduced several new tags to enhance the structure and functionality of web pages. Some of the key new tags introduced in HTML5 include:

1. ***<header>:*** Represents the header of a section or page. Typically contains headings, logos, and navigation menus.
2. ***<nav>:*** Defines a navigation menu. It is used to group navigation links.
3. ***<article>:*** Represents a self-contained piece of content that could be distributed and reused independently. Commonly used for blog posts, news articles, and forum posts.
4. ***<section>:*** Represents a thematic grouping of content. It helps to organize content into different sections.
5. ***<aside>:*** Defines content that is tangentially related to the content around it, such as sidebars or pull quotes.
6. ***<figure> and <figcaption>:*** Used together to represent any content that is referenced from the main content, such as images, charts, or videos, along with a caption.
7. ***<footer>:*** Represents the footer of a section or page. Often contains metadata, copyright information, and links to related documents.
8. ***<mark>:*** Highlights text within the context of its surrounding content, typically to denote search results or user input.
9. ***<progress>:*** Represents the completion progress of a task, such as the loading status of an image or the progress of a file upload.
10. ***<time>:*** Represents a specific period in time, either a point in time or a range.
11. ***<meter>:*** Represents a scalar measurement within a known range, such as disk usage or temperature.
12. ***<datalist>:*** Contains a set of <option> elements that represent the permissible or suggested options available to users in other controls.
13. ***<output>:*** Represents the result of a calculation or user action.
14. ***<canvas>:*** Provides a drawing space through scripting, allowing dynamic graphics and animations.
15. ***<audio> and <video>:*** These tags allow embedding audio and video content directly into a web page.

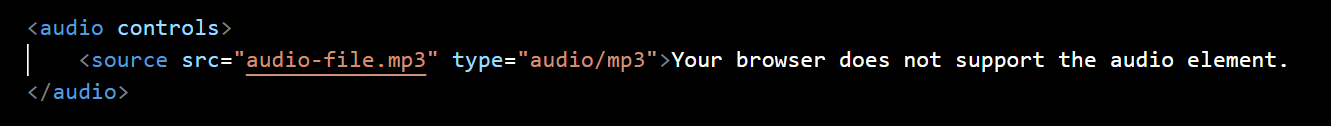
These tags, among others, contribute to making HTML5 more versatile and capable of supporting modern web development needs. Keep in mind that HTML specifications may evolve, so it's a good idea to refer to the latest documentation for the most up-to-date information.

**(Q.2) How to embed audio and video in a webpage?**

**(ANS):**

To embed audio and video in a webpage, you can use the <audio> and <video> tags, respectively. Here's a basic example for each:

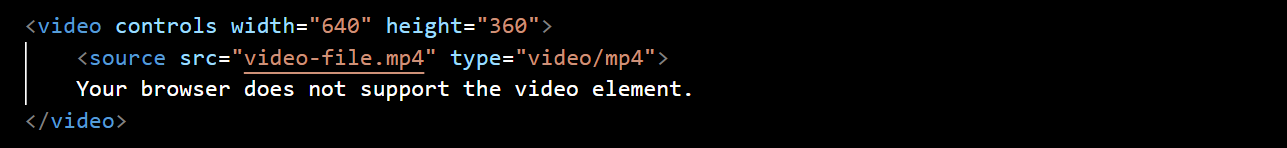
***Embedding Audio:***



In this example:

* The controls attribute adds audio playback controls (play, pause, volume, etc.).
* The <source> element provides different sources of the audio file, allowing the browser to choose the appropriate one based on compatibility.

***Embedding Video:***



In this example:

* The controls attribute adds video playback controls.
* The <source> element provides different sources of the video file, allowing the browser to choose the appropriate one based on compatibility.
* The width and height attributes set the dimensions of the video player.

You should replace "audio-file.mp3" and "video-file.mp4" with the actual paths to your audio and video files. Additionally, you can provide multiple <source> elements with different file formats to ensure compatibility across various browsers.

The text inside the <audio> and <video> tags ("Your browser does not support the...") will be displayed if the browser doesn't support the audio or video element.

Remember to adjust the dimensions, file types, and other attributes based on your specific requirements and the characteristics of your media files.

**(Q.3) Semantic element in HTML5?**

**(ANS):**

Semantic elements in HTML5 are tags that carry meaning about the structure and content of a web page. They provide additional information about the parts they enclose, making it clearer and more meaningful for both browsers and developers. Using semantic elements helps improve accessibility, search engine optimization (SEO), and the overall structure of your HTML document. Here are some key semantic elements introduced in HTML5:

1. ***<article>:*** Represents a self-contained piece of content that could be distributed and reused independently, such as a news article or blog post.
2. ***<section>:*** Defines a thematic grouping of content. It is used to group related content together, and it often corresponds to a section within a document.
3. ***<nav>:*** Represents a navigation menu. It is used to define a navigation block within a document, containing links to other pages or sections.
4. ***<header>:*** Represents the header of a section or page. It typically contains headings, logos, and navigation menus.
5. ***<footer>:*** Represents the footer of a section or page. It often contains metadata, copyright information, and links to related documents.
6. ***<aside>:*** Represents content that is tangentially related to the content around it. It is often used for sidebars, pull quotes, or related links.
7. ***<main>:*** Represents the main content of the document. It helps to identify the primary content within a document and is often used to exclude repetitive content like headers and footers.
8. ***<figure> and <figcaption>:*** Used together to represent any content that is referenced from the main content, such as images, charts, or videos, along with a caption.
9. ***<mark>:*** Highlights text within the context of its surrounding content, typically to denote search results or user input.
10. ***<time>:*** Represents a specific period in time, either a point in time or a range.
11. ***<progress>:*** Represents the completion progress of a task, such as the loading status of an image or the progress of a file upload.
12. ***<meter>:*** Represents a scalar measurement within a known range, such as disk usage or temperature.

These semantic elements help developers structure their HTML in a more meaningful way, making it easier to understand and maintain. They also provide valuable information to assistive technologies for better accessibility and contribute to improved SEO by conveying the hierarchical structure of the content.

**(Q.4) Canvas and SVG tags**

**(ANS):**

The <canvas> and <svg> tags are both used for creating graphics on web pages, but they have different approaches and use cases.

***<canvas> Tag:***

The <canvas> element is used for drawing graphics, animations, and other visual elements using JavaScript. It provides a bitmap drawing surface where you can use JavaScript to draw shapes, images, and manipulate pixels directly.

Here's a simple example:

A computer screen with text on it

Description automatically generated

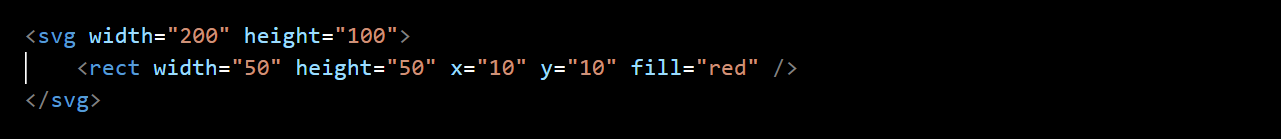
In this example:

* The <canvas> element creates a drawing area with an id of "myCanvas" and dimensions of 200 pixels by 100 pixels.
* The JavaScript code fetches the canvas element, gets its 2D rendering context, and then draws a blue rectangle on the canvas.

***<svg> Tag:***

The <svg> (Scalable Vector Graphics) element is an XML-based vector image format. It is used for describing two-dimensional vector graphics. SVG graphics can be created and edited with any text editor and scaled without losing quality.

Here's a simple example:



In this example:

* The <svg> element creates an SVG drawing area with dimensions of 200 pixels by 100 pixels.
* The <rect> element within the SVG draws a red rectangle with a width of 50 pixels, a height of 50 pixels, and positioned at (10, 10).

***Comparison:***

* Canvas is raster-based, meaning it works with pixels. Any drawings on the canvas are essentially a static image. It's suitable for complex graphics, animations, and pixel-based manipulations.
* SVG is vector-based, representing graphics as mathematical equations rather than pixels. It's suitable for graphics that need to scale without losing quality. SVG elements are part of the DOM, making them scriptable and accessible.

Choosing between <canvas> and <svg> depends on the specific requirements of your project. If you need pixel-level control and dynamic content, <canvas> may be more suitable. If you need scalable, resolution-independent graphics with easier interactivity and accessibility, <svg> is a better choice.