***Frontend Assignment***

***MODULE: 5 (Website Designing -*HTML5*)***

1. What are the new tags added in HTML5?

HTML5 introduced several new tags that offer improved semantics and enhance the structure of web documents. Here are some of the notable new tags introduced in HTML5:

1. `<header>`: Represents the introductory content or a container for the header section of a document or section.

2. `<nav>`: Defines a section containing navigation links.

3. `<article>`: Represents a self-contained composition within a document, such as a blog post, news article, or forum post.

4. `<section>`: Defines a generic section of a document or application, which could be a chapter, tabbed content, or any other distinct grouping.

5. `<aside>`: Represents a section of content that is indirectly related to the main content and can be considered as a sidebar or a separate block.

6. `<figure>`: Specifies self-contained content, such as images, diagrams, photos, code snippets, etc., that is referenced from the main content.

7. `<figcaption>`: Provides a caption or description for the content within a `<figure>` element.

8. `<main>`: Indicates the main content of a document. There should be only one `<main>` element per page.

9. `<footer>`: Represents the footer section of a document or a section. It often contains information about the author, copyright notice, or links to related documents.

10. `<time>`: Specifies a date, time, or duration. It can be used to mark up dates and times within the content.

11. `<mark>`: Highlights or emphasizes a specific portion of text within a document.

12. `<progress>`: Displays the progress of a task or process, typically shown as a progress bar.

13. `<meter>`: Represents a scalar measurement within a known range, such as disk usage, completion percentage, etc.

14. `<datalist>`: Provides a list of predefined options for input controls like `<input>` or `<select>`.

15. `<details>`: Creates a disclosure widget that can be used to show or hide additional content.

16. `<summary>`: Defines a visible heading for the `<details>` element.

1. How to embed audio and video in a webpage?

To embed audio and video in a webpage, HTML5 introduced the <audio> and <video> elements. Here's how you can use them:

1.Embedding Audio:

<audio controls>

<source src="audio-file.mp3" type="audio/mpeg">

<source src="audio-file.ogg" type="audio/ogg">

Your browser does not support the audio element.

</audio>

The <audio> element is used to embed audio content on a webpage.

The controls attribute adds a default set of playback controls to the audio player.

The <source> elements specify different audio sources in different formats (mp3 and ogg in this example).

The text "Your browser does not support the audio element." is displayed if the browser doesn't support the <audio> element.

2.Embedding Video:

<video controls width="400">

<source src="video-file.mp4" type="video/mp4">

<source src="video-file.webm" type="video/webm">

Your browser does not support the video element.

</video>

* The <video> element is used to embed video content on a webpage.
* The controls attribute adds a default set of playback controls to the video player.
* The width attribute specifies the width of the video player (400 pixels in this example).
* The <source> elements specify different video sources in different formats (mp4 and webm in this example).
* The text "Your browser does not support the video element." is displayed if the browser doesn't support the <video> element.

1. Semantic element in HTML5?

HTML5 introduced a set of semantic elements that provide a clearer structure and meaning to the content of a webpage. These semantic elements help search engines, assistive technologies, and developers understand the purpose and context of different sections of a document. Here are some of the key semantic elements in HTML5:

1. `<header>`: Represents the introductory content or a container for the header section of a document or section.

2. `<nav>`: Defines a section containing navigation links.

3. `<main>`: Indicates the main content of a document. There should be only one `<main>` element per page.

4. `<article>`: Represents a self-contained composition within a document, such as a blog post, news article, or forum post.

5. `<section>`: Defines a generic section of a document or application, which could be a chapter, tabbed content, or any other distinct grouping.

6. `<aside>`: Represents a section of content that is indirectly related to the main content and can be considered as a sidebar or a separate block.

7. `<footer>`: Represents the footer section of a document or a section. It often contains information about the author, copyright notice, or links to related documents.

8. `<figure>`: Specifies self-contained content, such as images, diagrams, photos, code snippets, etc., that is referenced from the main content.

9. `<figcaption>`: Provides a caption or description for the content within a `<figure>` element.

10. `<time>`: Specifies a date, time, or duration. It can be used to mark up dates and times within the content.

11. `<mark>`: Highlights or emphasizes a specific portion of text within a document.

12. `<progress>`: Displays the progress of a task or process, typically shown as a progress bar.

13. `<meter>`: Represents a scalar measurement within a known range, such as disk usage, completion percentage, etc.

14. `<details>`: Creates a disclosure widget that can be used to show or hide additional content.

15. `<summary>`: Defines a visible heading for the `<details>` element.

1. Canvas and SVG tags

The <canvas> and <svg> tags are both HTML5 elements used for creating graphics and visual elements on webpages, but they differ in their underlying technologies and how they work. Here's a brief overview of each:

1.<canvas>:

The <canvas> element provides a drawing surface on which you can use JavaScript to render graphics, animations, and interactive elements.

It uses a bitmap-based approach, where you have direct control over the individual pixels on the canvas.

You can draw shapes, lines, text, images, and apply transformations using JavaScript's Canvas API.

The content drawn on the <canvas> element is not part of the document's DOM tree, so it's not directly accessible or manipulable using standard HTML or CSS.

The canvas is defined with a specific width and height, and you can dynamically update and redraw its content as needed.

Example usage:

<canvas id="myCanvas" width="500" height="300"></canvas>

2. <svg>:

The <svg> element is used to define vector-based graphics directly within an HTML document.

SVG stands for Scalable Vector Graphics, which means the graphics are defined using XML-based markup.

SVG graphics are resolution-independent and can be scaled and resized without losing image quality.

You can create shapes, lines, curves, text, and apply styling and transformations using SVG's XML syntax or CSS.

The content of the <svg> element is part of the document's DOM tree, so it can be styled and manipulated using CSS and JavaScript like any other HTML element.

SVG supports interactivity, animations, and scripting using JavaScript.

Example usage:

<svg id="mySvg" width="500" height="300">

<circle cx="50" cy="50" r="30" fill="red" />

<rect x="100" y="50" width="100" height="100" fill="blue" />

</svg>