1. What are the new tags added in HTML5?

HTML5 introduced several new tags and elements to enhance the structure and functionality of web documents. Here are some of the key new tags added in HTML5:

1. **<header>** : Represents a container for introductory content or a set of navigational links. It often includes elements like headings, logos, and navigation menus.

2. **<nav>** : Defines a section of a document that contains navigation links. Typically used for menus, lists of links, or other navigation-related content.

3. **<main>** : Specifies the main content of a document, excluding headers, footers, and sidebars. There should be only one <main> element per page.

4. **<article>** : Represents a self-contained piece of content that can be distributed and reused independently, such as a news article, blog post, or forum post.

5. **<section>** : Divides a document into sections or thematic groupings of content. It helps improve the structure and semantics of a web page.

6**. <aside>** : Defines content that is tangentially related to the content around it. Typically used for sidebars, advertisements, or other content not critical to the main content.

7. **<figure>** and **<figcaption>** : Used together to embed images, illustrations, diagrams, or multimedia content, with **<figcaption>** providing a caption or description for the content within the <figure> element.

8. **<details>** and **<summary>**: Create a disclosure widget that allows users to show or hide additional content. **<details>** contains the content, and **<summary>** provides the label or heading for the disclosure.

9. **<mark>** : Highlights text within a document to draw attention to it, often used for search results or highlighting specific portions of content.

10. **<time>** : Represents a specific point in time or a duration. It can be used for dates, times, and time intervals, and it can include machine-readable date and time information.

11. **<meter>** : Defines a scalar measurement within a known range, such as a progress bar or gauge. It allows you to represent data visually.

12. **<progress>** : Represents the completion progress of a task or process, such as file uploads, downloads, or form submissions.

13. **<datalist>** and **<option>** : Used to create a set of predefined options for an <input> element with the list attribute, creating a dropdown-like interface for user input.

14. **<output>** : Displays the result of a calculation or user action, often used with form elements to provide feedback to users.

15. **<canvas>** : Provides a drawing surface for graphics and animations. JavaScript is commonly used to draw and manipulate content within the <canvas> element.

16**. <audio>** and **<video>** : Embed audio and video content in web pages. These elements support various multimedia formats and provide built-in playback controls.

17. **<source>** : Used within **<audio>** and **<video>** elements to specify multiple media resources, allowing the browser to choose the most suitable format based on compatibility.

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

Embedding Audio:

1. Choose an Audio File : First, you need an audio file in a supported format like MP3, WAV, or Ogg. Make sure you have the audio file ready.

2. Use the <audio> Element : Place the <audio> element in your HTML where you want the audio to appear. You can specify the source of the audio using the src attribute and provide alternative text in case the audio cannot be played using the <source> element.

```
 <audio controls>
  <source src="your-audio-file.mp3" type="audio/mpeg">
  Your browser does not support the audio element.
</audio>
```

- The controls attribute adds audio playback controls like play, pause, and volume.
- Inside the <audio> element, you can have multiple <source> elements with different audio formats to ensure cross-browser compatibility.

Embedding Video:

1. Choose a Video File: Similarly, you need a video file in a supported format like MP4, WebM, or Ogg. Have your video file ready.

2. Use the <video> Element: Place the <video> element in your HTML where you want the video to appear. You can specify the video source using the src attribute and provide fallback content inside the element.

Example :

```
<video controls>
  <source src="your-video-file.mp4" type="video/mp4">
  Your browser does not support the video element.
</video>
```

- The controls attribute adds video playback controls.
- Like with audio, you can include multiple <source> elements with different video formats for better compatibility.

3. Semantic element in HTML5?

1**. <header>** : Represents a container for introductory content or a set of navigational links. It typically includes elements like headings, logos, and navigation menus.

2. **<nav>** : Defines a section of a document that contains navigation links. It's often used for menus, lists of links, or other navigation-related content.

3. **<main>** : Specifies the main content of a document, excluding headers, footers, and sidebars. There should be only one <main> element per page.

4. **<article>** : Represents a self-contained piece of content that can be distributed and reused independently, such as a news article, blog post, or forum post.

5**. <section>** : Divides a document into sections or thematic groupings of content. It helps improve the structure and semantics of a web page.

6. **<aside>** : Defines content that is tangentially related to the content around it. It's typically used for sidebars, advertisements, or other content not critical to the main content.

7**. <figure>** and **<figcaption>**: Used together to embed images, illustrations, diagrams, or multimedia content, with **<figcaption>** providing a caption or description for the content within the <figure> element.

8. **<footer>** : Represents the footer of a document or a section. It often contains metadata, copyright information, contact details, or links to related documents.

9. **<time>** : Represents a specific point in time or a duration. It can be used for dates, times, and time intervals, and it can include machine-readable date and time information.

10. **<mark>** : Highlights text within a document to draw attention to it, often used for search results or highlighting specific portions of content.

11. <**meter>** : Defines a scalar measurement within a known range, such as a progress bar or gauge. It allows you to represent data visually.

12. **<progress> :** Represents the completion progress of a task or process, such as file uploads, downloads, or form submissions.

13. **<details>** and **<summary>** : Create a disclosure widget that allows users to show or hide additional content. **<details>** contains the content, and <summary> provides the label or heading for the disclosure.

14. **<datalist>** and **<option>** : Used to create a set of predefined options for an **<input>** element with the list attribute, creating a dropdown-like interface for user input.

15. **<output>** : Displays the result of a calculation or user action, often used with form elements to provide feedback to users.

4. Canvas and SVG tags

Both the <canvas> and <svg> tags in HTML5 are used for rendering graphics on web pages, but they differ in how they create and manipulate graphics.

1. <canvas> Tag:

- The <canvas> element is used to create graphics using JavaScript. It provides a bitmap-based drawing surface where you can use JavaScript to draw shapes, images, and animations.

- It is ideal for rendering dynamic and interactive graphics, such as games, charts, and animations.

- The graphics drawn on a <canvas> element are not part of the document's DOM (Document Object Model), which means you cannot select or manipulate individual shapes or elements using standard HTML or CSS.

- <canvas> is well-suited for situations where you need fine control over graphics rendering and animation, and it's typically used with JavaScript libraries like HTML5 Canvas API.

Example usage of <canvas>:

```
<canvas id="myCanvas" width="400" height="200"></canvas>
```

```
<script>
  var canvas = document.getElementById("myCanvas");
  var context = canvas.getContext("2d");

  context.fillStyle = "blue";
  context.fillRect(50, 50, 100, 100);
</script>
```

2. <svg> Tag:

- The <svg> element is used to create vector graphics that are part of the document's DOM. SVG stands for Scalable Vector Graphics, and it uses XML syntax to define shapes and graphics.
- SVG graphics can be manipulated using JavaScript, just like HTML elements, because they are part of the DOM.
- Unlike <canvas>, SVG allows you to select and manipulate individual shapes and elements using CSS and JavaScript.

Example usage of <svg> :

```
<svg width="400" height="200">
  <rect x="50" y="50" width="100" height="100" fill="blue" />
</svg>
```