Q1. What are the new tags added in HTML5?

Ans: <article>: Represents a self-contained composition that can be independently distributed or reused.

<aside>: Defines content aside from the content it is placed in, such as sidebars or tangentially related content.

<audio>: Embeds sound content in a document, such as music or other audio streams.

<canvas>: Provides a space in which you can dynamically render graphics, charts, or other visual images.

<datalist>: Specifies a list of pre-defined options for input controls.

<details>: Represents a disclosure widget from which the user can obtain additional information or controls.

<figcaption>: Represents a caption or a legend for a figure element.

<figure>: Represents self-contained content, such as illustrations, diagrams, photos, code listings, etc.

<footer>: Defines a footer for a document or section.

<header>: Represents introductory content, usually at the beginning of a section or a page.

<main>: Specifies the main content of a document.

<mark>: Defines text which should be marked or highlighted.

<nav>: Defines a set of navigation links.

<section>: Defines a section in a document, such as chapters, headers, footers, or any other sections of the document.

<time>: Represents a specific period in time or a range of time.

These new tags and attributes help to provide better semantic meaning to the content of a web page, making it more accessible to search engines and assistive technologies, and improving the overall structure of the web page.

Q2. How to embed audio and video in a webpage?

Ans: 1. Embedding Audio:

<audio controls>

<source src="path\_to\_audio\_file.mp3" type="audio/mpeg">

Your browser does not support the audio tag.

</audio>

In this example, replace "path\_to\_audio\_file.mp3" with the actual path to your audio file. You can also use other audio formats such as WAV or OGG by changing the "type" attribute accordingly.

2. Embedding Video:

<video width="320" height="240" controls>

<source src="path\_to\_video\_file.mp4" type="video/mp4">

Your browser does not support the video tag.

</video>

Replace "path\_to\_video\_file.mp4" with the actual path to your video file. You can also use other video formats such as WebM or OGG by changing the "type" attribute accordingly.

Both the <audio> and <video> elements support various attributes that you can use to customize the appearance and behavior of the media player. The "controls" attribute in both examples adds basic playback controls to the audio or video player.

Make sure that the file paths are correct and that the web server is configured to serve audio and video files correctly. Additionally, ensure that the browser supports the audio and video formats you're using.

You can also use JavaScript frameworks like React or Vue.js, along with HTML5's video and audio elements, to create more sophisticated media player interfaces.

Q3. Semantic element in HTML5?

Ans: HTML5 introduced several new semantic elements that provide a more descriptive structure to web documents. These elements help define the different parts of a web page more accurately, making it easier for search engines and screen readers to interpret the content. Some of the important semantic elements introduced in HTML5 are:

<header>: Represents introductory content at the beginning of a section or webpage. It typically contains a group of introductory or navigational aids.

<nav>: Defines a section of a page that contains navigation links, such as menus, tables of contents, and indexes.

<main>: Represents the main content of the document. It should not contain content that is repeated across multiple web pages, such as navigation links and footers.

<section>: Defines a thematic grouping of content, typically with a heading. It's often used to group together related content on a webpage.

<article>: Represents a self-contained composition in a document, such as a blog post, article, or news story.

<aside>: Defines a section of a page that contains content related to the main content, but which is not essential to understanding the main content.

<footer>: Represents a footer for its nearest section or the nearest ancestor sectioning content or sectioning root element. It typically contains information about the author, copyright details, links to privacy policy, contact information, and similar content.

<figure> and <figcaption>: Used together to represent self-contained content, such as images, diagrams, photos, code snippets, and more, along with a caption that describes the content.

Using these semantic elements in your HTML code not only helps to structure your webpage in a more meaningful way but also enhances accessibility and search engine optimization (SEO) for your website.

Q4. Canvas and SVG tags

Ans:

The HTML <canvas> and <svg> elements are both used to create graphics and visualizations on web pages, but they have different underlying structures and use cases.

<canvas> Element:

The <canvas> element is used to draw graphics, animations, and other visual images on the fly using JavaScript. It provides a rectangular area where you can use JavaScript to draw anything, such as graphs, animations, games, and more. It essentially provides a bitmap area that you can manipulate through JavaScript's drawing functions. Here's an example: <canvas id="myCanvas" width="200" height="100"></canvas>

You can use JavaScript to draw on the canvas: var canvas = document.getElementById('myCanvas');

var ctx = canvas.getContext('2d');

ctx.fillStyle = 'rgb(200, 0, 0)';

ctx.fillRect(10, 10, 50, 50);

**<svg>** Element: The **<svg>** element is used to define graphics for the web in XML format. It allows you to create shapes and images using XML-based markup and is widely used for creating vector graphics, such as logos, icons, and illustrations. You can include the **<svg>** element directly within your HTML document. Here's a basic example: <svg width="100" height="100">

<circle cx="50" cy="50" r="40" stroke="black" stroke-width="3" fill="red" />

</svg>

You can also use the <svg> element to include more complex graphics and shapes using XML-based syntax. It's possible to manipulate SVG elements with JavaScript to create interactive and dynamic visualizations.

In summary, the <canvas> element is suitable for rendering bitmap-based graphics, animations, and games, while the <svg> element is suitable for creating scalable vector graphics and illustrations. The choice between using <canvas> and <svg> depends on the specific requirements of your project, the type of graphics you want to create, and the level of interactivity and animation needed.