**Module (HTML5) – 3**

**Que.1 What are the new tags added in HTML5?**

**Ans**. HTML5 introduced several new tags that enhance the structure and functionality of web pages. Some of the notable new tags in HTML5 include:  
  
1. `<header>`: Represents the introductory content or a group of navigational links at the top of a document or section.  
2. `<nav>`: Defines a section of navigation links.  
3. `<section>`: Represents a standalone section of content within a document.  
4. `<article>`: Defines a self-contained composition that can be independently distributed or syndicated.  
5. `<aside>`: Represents content that is tangentially related to the main content, such as sidebars or pull quotes.  
6. `<footer>`: Represents the footer of a document or section, typically containing information about the author, copyright, or contact details.  
7. `<figure>` and `<figcaption>`: Used together to represent self-contained content, such as images or diagrams, along with their captions.  
8. `<video>` and `<audio>`: Tags for embedding video and audio content, respectively.  
9. `<canvas>`: Provides a space for dynamic, scriptable rendering of graphics or images.  
10. `<progress>`: Represents the progress of a task or process.  
11. `<meter>`: Represents a scalar measurement within a known range.  
12. `<datalist>`: Specifies a list of pre-defined options for input controls, such as `<input>` or `<select>`.  
13. `<time>`: Represents a specific time or a range of time.  
  
These are just a few examples of the new tags introduced in HTML5.

**Que.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 in HTML5. Here's a step-by-step guide:  
  
1. Prepare your audio or video file: Make sure you have the audio or video file in a compatible format (such as MP3, WAV, or MP4) and accessible on your web server.  
  
2. Add the `<audio>` or `<video>` tag: In your HTML file, add the appropriate tag depending on whether you want to embed audio or video.  
  
For audio:

**<audio src="path/to/audio-file.mp3" controls></audio>**

For vieo:

**<video src="path/to/video-file.mp4" controls></video>**

Replace "path/to/audio-file.mp3" and "path/to/video-file.mp4" with the actual paths to your audio and video files.  
  
3. Customize the playback options: You can add additional attributes to the `<audio>` or `<video>` tag to customize the playback options. For example, you can include the `controls` attribute to display playback controls, such as play, pause, and volume controls.  
  
4. Provide alternative content: It's a good practice to provide alternative content in case the browser doesn't support the `<audio>` or `<video>` tag or the specified file format. You can do this by placing text or a link inside the `<audio>` or `<video>` tags, which will be displayed if the browser doesn't support the tag.

**<audio src="path/to/audio-file.mp3" controls>**

**Your browser does not support the audio element.**

**You can download the audio file <a href="path/to/audio-file.mp3">here</a>.**

**</audio>**

**<video src="path/to/video-file.mp4" controls>**

**Your browser does not support the video element.**

**You can download the video file <a href="path/to/video-file.mp4">here</a>.**

**</video>**

5. Save and test: Save your HTML file and open it in a web browser to see the embedded audio or video. You should be able to play, pause, and control the playback using the provided controls.  
  
Remember to replace "path/to/audio-file.mp3" and "path/to/video-file.mp4" with the actual paths to your audio and video files.

**Que.3 Semantic element in HTML5?**

**Ans**. HTML5 introduced several semantic elements that provide meaning and structure to the content of a web page. These semantic elements help search engines, screen readers, and other technologies understand the purpose and context of different sections of a webpage. Some of the semantic elements in HTML5 include:  
  
1. `<header>`: Represents the introductory content or a group of navigational links at the top of a document or section.  
2. `<nav>`: Defines a section of navigation links.  
3. `<section>`: Represents a standalone section of content within a document.  
4. `<article>`: Defines a self-contained composition that can be independently distributed or syndicated.  
5. `<aside>`: Represents content that is tangentially related to the main content, such as sidebars or pull quotes.  
6. `<footer>`: Represents the footer of a document or section, typically containing information about the author, copyright, or contact details.  
7. `<main>`: Specifies the main content of a document, excluding header, footer, and navigation.  
8. `<figure>` and `<figcaption>`: Used together to represent self-contained content, such as images or diagrams, along with their captions.  
9. `<time>`: Represents a specific time or a range of time.  
10. `<mark>`: Highlights or denotes a portion of text for reference or emphasis.  
11. `<progress>`: Represents the progress of a task or process.  
12. `<meter>`: Represents a scalar measurement within a known range.  
13. `<datalist>`: Specifies a list of pre-defined options for input controls, such as `<input>` or `<select>`.  
14. `<summary>` and `<details>`: Used together to create an interactive disclosure widget for showing or hiding additional content.  
  
These semantic elements provide a more meaningful and structured way to organize and present content on a webpage, improving accessibility and search engine optimization.

**Que.4 Canvas and SVG tags**

**Ans.** Both the `<canvas>` and `<svg>` tags are used in HTML5 to create and display graphics, but they have different approaches and use cases.  
  
The `<canvas>` tag provides a drawing surface on which you can use JavaScript to dynamically render graphics, animations, and interactive elements. It offers a bitmap-based approach, where you manipulate individual pixels to create visuals. The canvas element is ideal for complex, dynamic graphics and animations, such as games or data visualizations.  
  
On the other hand, the `<svg>` tag stands for Scalable Vector Graphics. It is a markup language that describes two-dimensional vector graphics. SVG graphics are resolution-independent and can be scaled without losing quality. The SVG tag allows you to define shapes, paths, text, and other graphical elements using XML-based syntax. SVG is well-suited for static or interactive graphics, icons, logos, and illustrations.  
  
In summary, `<canvas>` is suitable for bitmap-based graphics and animations created with JavaScript, while `<svg>` is used for scalable vector graphics defined with XML-based syntax. The choice between the two depends on the specific requirements of your project and the type of graphics you want to create.