semantic and non-semantic tags =>

Semantic HTML tags are tags that define the meaning of the content they contain. For example, tags like <header>, <article>, and <footer> are semantic HTML tags. They clearly indicate the role of the content they contain.

Non-semantic elements: Unlike semantic elements, they don't have any meaning. They don't tell anything about the content they contain. They can be used with different attributes to mark up semantics common to a group.

Following is the list of some semantic elements :

- article
- aside
- details
- figcaption
- figure
- footer
- form
- header
- main
- mark
- nav
- table
- section

Following is the list of some non-semantic elements:

- div
- span

 tag: The div tag is known as Division tag. The div tag is used in HTML to make divisions of content on the web page like (text, images, header, footer, navigation bar, etc). Div tag has both opening( ) and closing ( ) tags and it is mandatory to close the tag. As we know Div tag is a block-level tag. In this example, the div tag contains the entire width. It will be displayed div tag each time on a new line, not on the same line.

```
<body>
   <div> div tag </div>
   <div> div tag </div>
   <div> div tag </div>
   <div> div tag </div>
</body>
```

 tag: The HTML span element is a generic inline container for inline elements and content. It used to group elements for styling purposes (by using the class or id attributes). A better way to use it when no other semantic element is available. The span tag is very similar to the div tag, but div is a block-level tag and span is an inline tag.

```
   <p><span style="background-color:lightgreen">
      GeeksforGeeks</span> is A Computer Science Portal
      where you can<span style="color:blue;">
      Publish</span> your own <span
      style="background-color:lightblue;">articles</span>
       and share your knowledge with the world!!
   </p>
```

Top 12 HTML5 Features, which are explained in detail below:

Semantic Elements

Audio and Video Support

Canvas Elements => Canvas Elements is a top-notch feature that has made the tedious task of handling graphics easier for developers. With the help of Canvas elements, you can easily draw graphics using JavaScript. It is optimum for creating simple animations and drawing photo compositions.

Geolocation API => The Geolocation API is an HTML feature that is used to access the geographical position of a user, however, it is not accessed unless the user approves of it. If you're wondering where this feature is useful, these come in handy while creating apps like taxi apps, food order tracking apps, fitness tracking apps, and more.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Location</title>
</head>
<body>
    <h1>Geolation</h1>
    <button id="getLocation">Click on me</button>
</body>
<script>
    var getLoc = document.getElementById("getLocation");
    getLoc.addEventListener('click',event=>{
        if('getlocation' in navigator){
            navigator.geolocation.getCurrentPosition(pos=>{
            let latitude = pos.coords.latitude;
            let longitude = pos.coords.longitude;

            console.log("the latitude is",latitude);
            console.log("the longitude is",longitude);
            console.log(latitude, longitude)
        },error=>{
            console.log("Denied by user",error.code)
        })
    }else{
        console.log("geolocation not supported");
    }
    })
</script>
</html>
```

Local Storage => It is a modern feature of HTML and several browsers that typically store data in the user's browsers and can access them with the help of JavaScript APIs

Responsive Images => Earlier to create responsive images on the web, you have to rely upon several lines of CSS and sometimes JavaScript, however, HTML 5 makes the process handy by including srcset attribute to specify multiple versions of an image at different screen resolutions.

<img

alt="image alt text" src="rose.jpg" srcset=" rose.jpg 240w, rose.jpg 300w, rose.jpg 720w "

sizes=" (min-width: 960px) 720px, 100vw ">

Web Workers => When you are performing several demanding tasks at once, the browser gets sluggish and responds slowly. Because of this, web workers were added to HTML 5 to allow scripts to operate in the background without interfering with the UI thread.

Drag and Drop API => Drag and drop is among the most unique features of HTML5 that allow you to grab any element in the DOM and drop it to a different location. To create an element able to drag and drop, set the attribute "draggable" on the tag and put its value to true. Let's understand this process using a practical coding example.

Form Enhancements => HTML 5 introduces new features for your existing forms on HTML to create a more robust user experience. Some of the key enhancements in the new HTML 5 include new input times such as email, URL, and more, placeholder text, required fields feature, validation, and more.

Web Sockets => In the previous versions of HTML, when a client sends a request to the backend server, the server then responds afterward. However, in HTML 5 we can establish a bidirectional live communication between the server and the client (a web browser) to reduce the latency in the responses.

Micro Data => Micro Data in simpler words is a further deeper level to provide semantics to your webpage.

Cross Document Messaging => In general, web browsers don't let web pages from different domains influence each other, this is done for several security reasons, however, even if the web pages don't intend to intrigue the privacy or harm the other web page. There are several scenarios, in which you want to access cross-document communication to make your web page more interactive with less effort.

New input types =>

13 new input types introduced in HTML5 form =>

color: This input type allows the user to select a color from a color picker.

<input type="color">

Select your favorite color:

<input type="color"

value="#009900"

id="color">

date: This input type allows the user to select a date from a drop-down calendar.

```
<input type="date">
<input type="date"
        id="test"
        value="2019-07-02T25:32Z">
```

time: This input type allows the user to enter a time.

```
<input type="time">
<form action="#">
      Input Time: <input type="time" name="time">

      <input type="submit" value="Submit">
</form>
```

datetime: This input type allows the user to select date and time along with timezone.

datetime-local: This input type allows the user to select both local date and time.

week: This input type allows the user to select week and year from the drop-down calendar.

```
<input type="week">
<form>
      <input type="week">
      <br>
      <br>
</form>
```

email: This input type allows the user to enter an e-mail address.

```
<input type="email">
<form>
         Email: <input type ="email"
            value="">
</form>
```

month: This input type allows the user to select a month and year from a drop-down calendar.

```
<input type="month">
```

```
<form id="NEW">
   <input type="month" id="month_id"
        name="month" >
</form>
```

number: This input type allows the user to enter a numerical value.

range: This input type allows the user to enter a numerical value within a specified range.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
    initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1>Display range</h1>
    <form action="">
        <input type="range" id="vol" min="0" max="100">
        <input type="submit">
    </form>
</body>
</html>
```

search: This input type allows the user to enter a search string within the input field.

```
<!DOCTYPE html>
<html>
<body>
<h1>Display a Search Field</h1>
<form action="/action_page.php">
  <label for="gsearch">Search Google:</label>
  <input type="search" id="gsearch" name="gsearch">
  <input type="submit">
</form>
</body>
</html>
```

tel: This input type allows the user to enter a telephone number.

```
<!DOCTYPE html>
<html>
<body>
<h1>Display a Telephone Input Field</h1>
<form action="/action_page.php">
  <label for="phone">Enter a phone number:</
  label><br><br>
  <input type="tel" id="phone" name="phone"
  placeholder="123-45-678" pattern="[0-9]{3}-
  [0-9]{2}-[0-9]{3}" required><br><br>
  <small>Format: 123-45-678</small><br><br>
  <input type="submit">
</form>
</body>
</html>
```

url: This input type allows the user to enter the URL. Uniform Resource Locator

```
<!DOCTYPE html>
<html>
<body>
<h1>Display a URL Input Field</h1>
<form action="/action_page.php">
  <label for="homepage">Add your homepage:</
  label>
  <input type="url" id="homepage"
  name="homepage"><br><br>
  <input type="submit">
</form>
</body>
</html>
```

HTML Canvas =>

The HTML “canvas” element is used to draw graphics via JavaScript. The “canvas” element is only a container for graphics. One must use JavaScript to actually draw the graphics. Canvas has several methods for drawing paths, boxes, circles, text, and adding images. The canvas would be a rectangular area on an HTML page. By default, a canvas has no border and no content.

Rectangular canvas area, you must add a JavaScript to do the drawing.

```
<body>
  <canvas id="myCanvas" width="200" height="100" style="border:1px solid #d3d3d3;">
    Your browser does not support the HTML canvas tag.</canvas>

    <script>
    var c = document.getElementById("myCanvas");
    var ctx = c.getContext("2d");
    ctx.moveTo(0,0);
    ctx.lineTo(200,100);
    ctx.stroke();
    </script>
</body>
```

Draw a Circle

```
<body>
  <canvas id="myCanvas" width="200" height="100" style="border:1px solid #d3d3d3;">
    Your browser does not support the HTML canvas tag.</canvas>

    <script>
      var c = document.getElementById("myCanvas");
      var ctx = c.getContext("2d");
      ctx.beginPath();
      ctx.arc(95, 50, 40, 0, 2 * Math.PI);
      ctx.stroke();
      </script>
</body>
```

Draw a Text

```
<body>
  <canvas id="myCanvas" width="200" height="100" style="border:1px solid #d3d3d3;">
    Your browser does not support the HTML canvas tag.</canvas>
    <script>
      var c = document.getElementById("myCanvas");
      var ctx = c.getContext("2d");
      ctx.font = "30px Arial";
      ctx.fillText("Hello World", 10, 50);
    </script>
</body>
```

10 is margin 50 is padding

Draw Linear Gradient

```
<canvas id="myCanvas" width="200" height="100" style="border:1px solid #d3d3d3;">
  Your browser does not support the HTML canvas tag.</canvas>
  <script>
    var c = document.getElementById("myCanvas");
    var ctx = c.getContext("2d");

    // Create gradient
    var grd = ctx.createLinearGradient(0, 0, 200, 0);
    grd.addColorStop(0, "red");
    grd.addColorStop(1, "white");

    // Fill with gradient
    ctx.fillStyle = grd;
    ctx.fillRect(10, 10, 150, 80);
    </script>
```

draw the image

```
<html>
<head>
  <title>
    HTML
  </title>
</head>
<body>
    <img id="image" src=
"https://www.insidehighered.com/sites/default/files/media/iStock-1012331444.jpg"
         alt="logo" width="250" height="200">
    <p>Canvas to fill:</p>
    <canvas id="gfg" width="300" height="300" style="border:1px solid #d3d3d3; ">
    </canvas>
    <p> <button onclick="gfg()"> Click to Try </button> </p>
    <script>
    function gfg() {
        let g = document.getElementById("gfg");
        let ks = g.getContext("2d");
        let img = document.getElementById("image");
        ks.drawImage(img, 0, 0);
    }
    </script>
```

Shadow blur property

```
<body>
  <canvas id="GFG"
  width="500"
  height="250" ;>
</canvas>
<script>
let g = document.getElementById("GFG");
let geeks = g.getContext("2d");
geeks.shadowBlur = 20;
geeks.shadowColor = "yellow";
geeks.fillStyle = "red";
geeks.fillRect(30, 20, 100, 80);
</script>
</body>
```

SVG =>

SVG stands for Scalable Vector Graphics. It basically defines vector-based graphics in XML format. SVG graphics do NOT lose any quality if they are zoomed or resized. Every element and every attribute in SVG files can be animated.

SVG line in HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
    initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <svg height="250" width="600">
        <line x1="10" y1="10" x2="400" y2="400"
              style="stroke:rgb(0,0,255);stroke-width:3" />
    </svg>
</body>
</html>
```

SVG Rectangle in HTML

```
</head>
<body>
  <svg width="400" height="100">
 <rect width="400" height="100" style="fill: rgb(0, 0, 255); stroke-width: 10;
            stroke: rgb(0, 0, 0)" />
</svg>
</body>
```

SVG Circle in HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
    initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <svg width="200" height="200">
        <circle cx="80" cy="80" r="50" stroke="black"
                stroke-width="2" fill="grey" />
    </svg>
</body>
</html>
```

SVG Star in HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
    initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <svg width="300" height="200">
        <polygon points="100,10 40,198 190,78 10,78 160,198"
                 style="fill: grey; stroke: orange;
                        stroke-width: 5; fill-rule:
                        evenodd" />
    </svg>
</body>
</html>
```

SVG Logo

```
<svg height="130" width="500">
  <defs>
    <linearGradient id="grad1" x1="0%" y1="0%" x2="100%" y2="0%">
      <stop offset="0%" style="stop-color:rgb(255,255,0);stop-opacity:1" />
      <stop offset="100%" style="stop-color:rgb(255,0,0);stop-opacity:1" />
    </linearGradient>
  </defs>
  <ellipse cx="100" cy="70" rx="85" ry="55" fill="url(#grad1)" />
  <text fill="#ffffff" font-size="45" font-family="Verdana" x="50" y="86">SVG</text>
  Sorry, your browser does not support inline SVG.
</svg>
```

Advantages of SVG: Advantages of using SVG over other image formats (like JPEG and GIF) are:

SVG images can be created and edited with any text editor.

SVG images can be printed with high quality at any resolution.

Differences between HTML SVG and HTML Canvas:

SVG is resolution independent whereas CANVAS is resolution-dependent.

SVG supports event handlers whereas CANVAS doesn't have support for event handlers.

HTML <audio> Tag

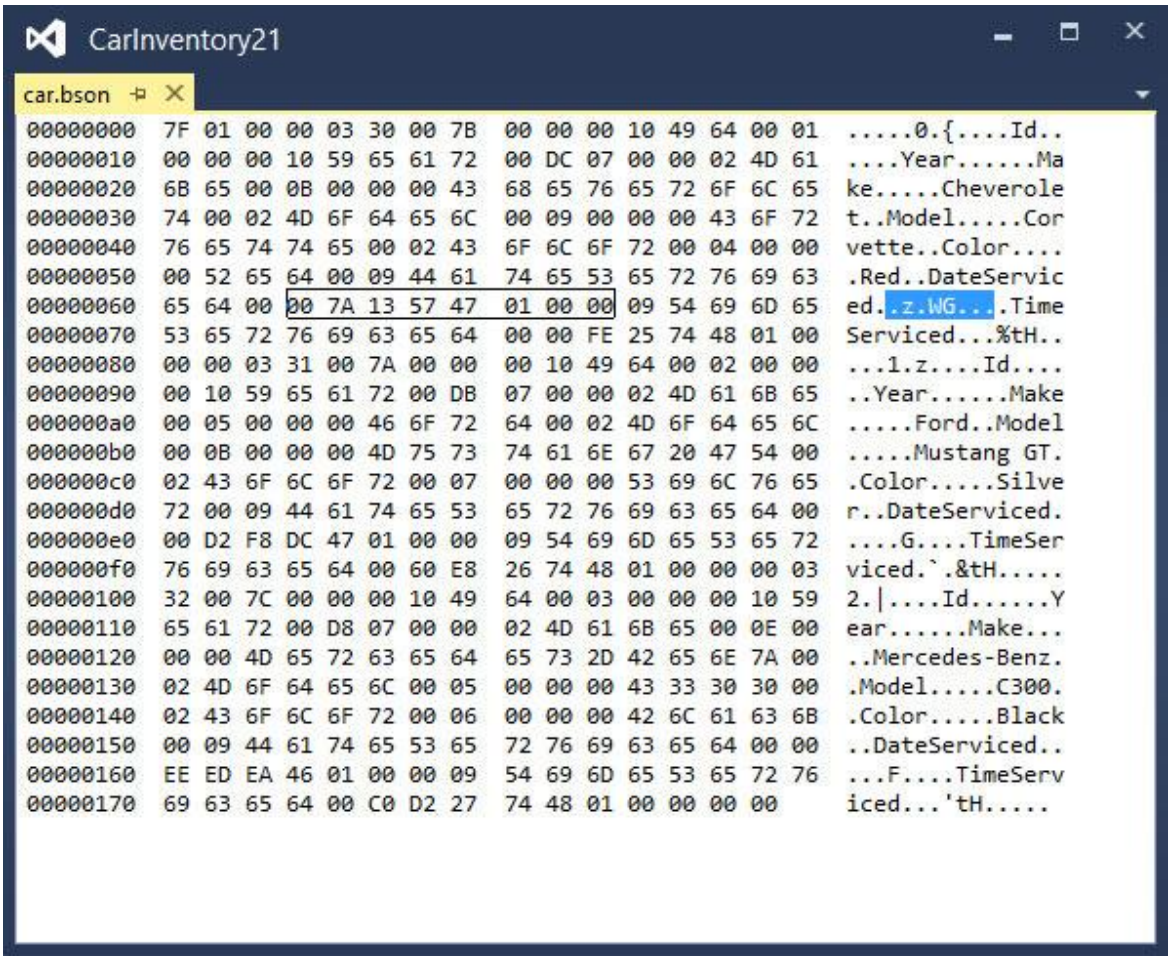
```
<!DOCTYPE html>
<html>
<body>
<h1>The audio element</h1>
<p>Click on the play button to play a sound:</p>
<audio controls>
  <source src="horse.ogg" type="audio/ogg">
  <source src="horse.mp3" type="audio/mpeg">
  Your browser does not support the audio element.
</audio>
</body>
</html>
```

HTML <video> Tag

```
<!DOCTYPE html>
<html>
<body>
<h1>The video element</h1>
<video width="320" height="240" controls>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
  Your browser does not support the video tag.
</video>
</body>
```

Attribute	Value	Description
autoplay	autoplay	Specifies that the video will start playing as soon as it is ready
controls	controls	Specifies that video controls should be displayed (such as a play/pause button etc).
height	*pixels*	Sets the height of the video player
loop	loop	Specifies that the video will start over again, every time it is finished
muted	muted	Specifies that the audio output of the video should be muted
poster	*URL*	Specifies an image to be shown while the video is downloading, or until the user hits the play button
preload	auto metadata none	Specifies if and how the author thinks the video should be loaded when the page loads
src	*URL*	Specifies the URL of the video file
width	*pixels*	Sets the width of the video player

BSON(binary encoded Javascript Object Notation (JSON)) is just binary JSON



JSON is a text-based, human-readable information compatibility organization utilized for speaking to basic information structures and objects in web browser-based code.

```
[
  {
    "date":"2013-11-05",
    "locations":{
      "United States":4,
      "Germany":8
    }
  },
  {
    "date":"2013-11-11",
    "locations":{
      "South Africa":9
    }
  },
  {
    "date":"2013-11-12",
    "locations":{
      "Japan":6
    }
  }
]
```

Advantages of JSON:

Faster

Structured Data

Readable:

JSON is perfect for storing temporary data. For example, temporary data can be user-generated data, such as a submitted form on a website. JSON can also be used as a serialization data format for any programming language to provide a high level of interoperability.