HTML audio Tag:

The HTML audio tag allows you to play audio files on a web page, such as music, recorded speech, or a sound effect. The element lets users control playback of the audio file, and supports MP3 and WAV formats.

Example:

```
<html>
<head>
<title>Using Audio Tag</title>
</head>
<body>
        <audio src="https://assets.codepen.io/6093409/drums.mp3" controls>
        </audio>

        <audio controls>
                <source src="C:\xampp\htdocs\phpprogramsexamples\Fire_Alarm.mp3">
        </audio>
</body>
</html>
```

Output:

Attributes of HTML Audio Tag:

Attribute	Description
controls	It defines the audio controls which is displayed with play/pause buttons.
autoplay	It specifies that the audio will start playing as soon as it is ready.
loop	It specifies that the audio file will start over again, every time when it is completed.
muted	It is used to mute the audio output.
src	It specifies the source URL of the audio file.

Example:

```
<audio src="https://assets.codepen.io/6093409/drums.mp3" controls autoplay></audio>

<audio src="https://assets.codepen.io/6093409/drums.mp3" controls loop></audio>

<audio src="https://assets.codepen.io/6093409/drums.mp3" controls loop></audio>

<audio src="https://assets.codepen.io/6093409/drums.mp3" controls muted></audio>
```

HTML Video Tag:

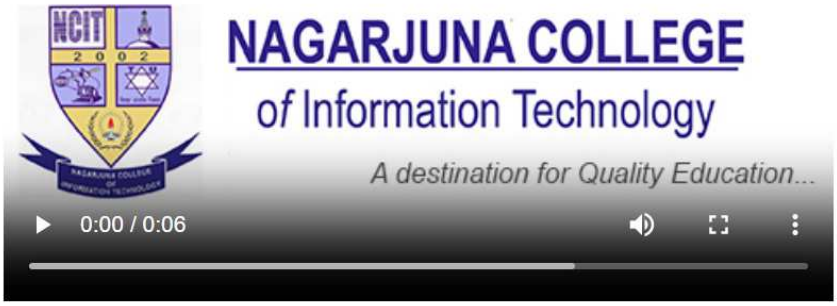
The HTML video tag is useful when you want to add a video to your web pages. In modern web browsers, adding the video is very easy. You do not need to install any extra plug-in for this. You can add the video by using a <video> tag.

Example:

```
<html>
<head>
<title>Using video Tag</title>
</head>
<body>
        <h2>This is an HTML video:</h2>
        <video controls poster="C:\xampp\htdocs\phpprogramsexamples\13913logo.jpg " width="520"
        height="250" src="https://assets.codepen.io/6093409/hubspot-video-example.mp4">
                This browser does not support HTML video.
        </video>

</body>
</html>
```

Output:

This is an HTML video:



Canvas:

HTML Canvas is a semantic element in HTML that provides a powerful tool for designing graphics on web pages. It is the only element container for graphics and offers various methods to draw and design boxes, circles, paths, text, images, etc. A rectangular area is created on an HTML page, with no border or constant, which can be accessed and manipulated using JavaScript.

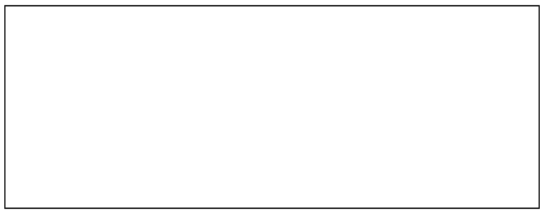
How does it work?

Canvas is a drawing surface that can be accessed and manipulated using JavaScript. When a web page loads, Canvas creates a blank rectangle on the page. The dimensions of this rectangle can be set using HTML attributes or JavaScript code. Once the rectangle is created, JavaScript code can draw shapes, lines, text, and images onto the Canvas surface

Example:

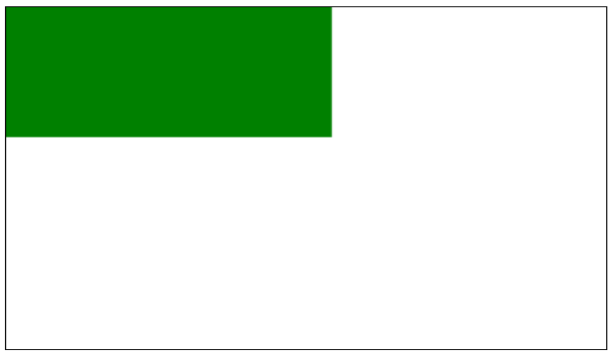
```
<html>
<head>
<title>HTML Canvas Tag</title>
</head>
<body>
        <canvas id="Canvas" width="300" height="100" style="border:1px solid;"></canvas>
</body>
</html>
```

Output:



```
<html>
<head>
<title>HTML Canvas Tag</title>
</head>
<body>
        <canvas id="myCanvas" width="350" height="250"
                style="border:1px solid;"> </canvas>
        <script>
                var canvas=document.getElementById('myCanvas');
                var ctx=canvas.getContext("2d");
                ctx.fillStyle ="green";
                ctx.fillRect(0,0,190,95);
        </script>
</body>
</html>
```

Output:



```
<html>
<head>
<title>Canvas Rect and Stroke</title> After rect(), we'll use the stroke() functionagain to draw the rectangle.
</head>
<body>
        <canvas id="myCanvas" width="" 300" height="200"
                style="border: 1px solid #d3d3d3"></canvas>

        <script>
                var canvas = document.getElementById("myCanvas");
                var context = canvas.getContext('2d');
                context.rect(20, 20, 200, 150);
                context.stroke();
        </script>
</body>
</html>
```

Output:

After rect(), we'll use the stroke() function again to draw the rectangle.

```
<html>
<head>
<title>Canvas Line</title>
</head>
<body>
        <canvas id="myCanvas" width="" 200" height="100"
                style="border: 1px solid #d3d3d3"></canvas>

        <script>
                 var canvas=document.getElementById('myCanvas');
                 var ctx=canvas.getContext("2d");
                ctx.moveTo(20,20);
                ctx.lineTo(200,100);
                ctx.stroke();
        </script>
</body>
</html>
```

Output:

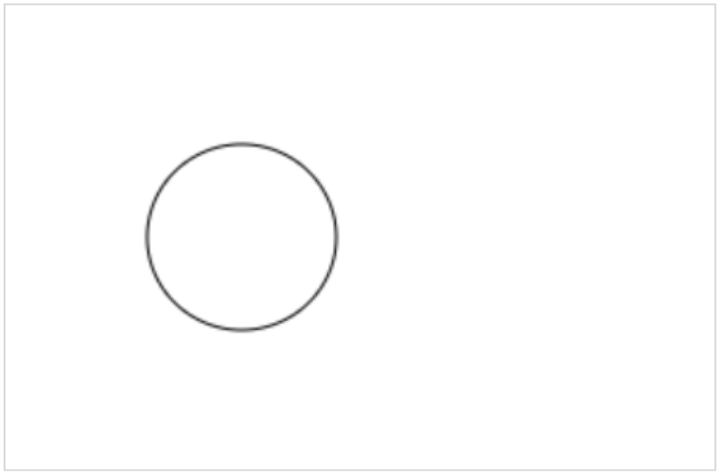
How to draw a circle on canvas?

- First of all, give begin path. We use the method ctx.beginPath();
- For creating the curve, we use the method ctx.arc(95,50,40,0,2*Maths.PI);
- Then we use the ink method to draw the circle ctx.stroke();

Example:

```
<html>
<head>
<title>Canvas Draw a Circle</title>
</head>
<body>
        <canvas id ="myCanvas" width=""200" height="200" style="border: 1px solid #d3d3d3">

        <script>
                var canvas=document.getElementById('myCanvas');
                var ctx=canvas.getContext("2d");
                ctx.beginPath();
                ctx.arc(100,100,40,0,2*Math.PI );
                ctx.stroke();;
        </script>
</body>
</html>
```

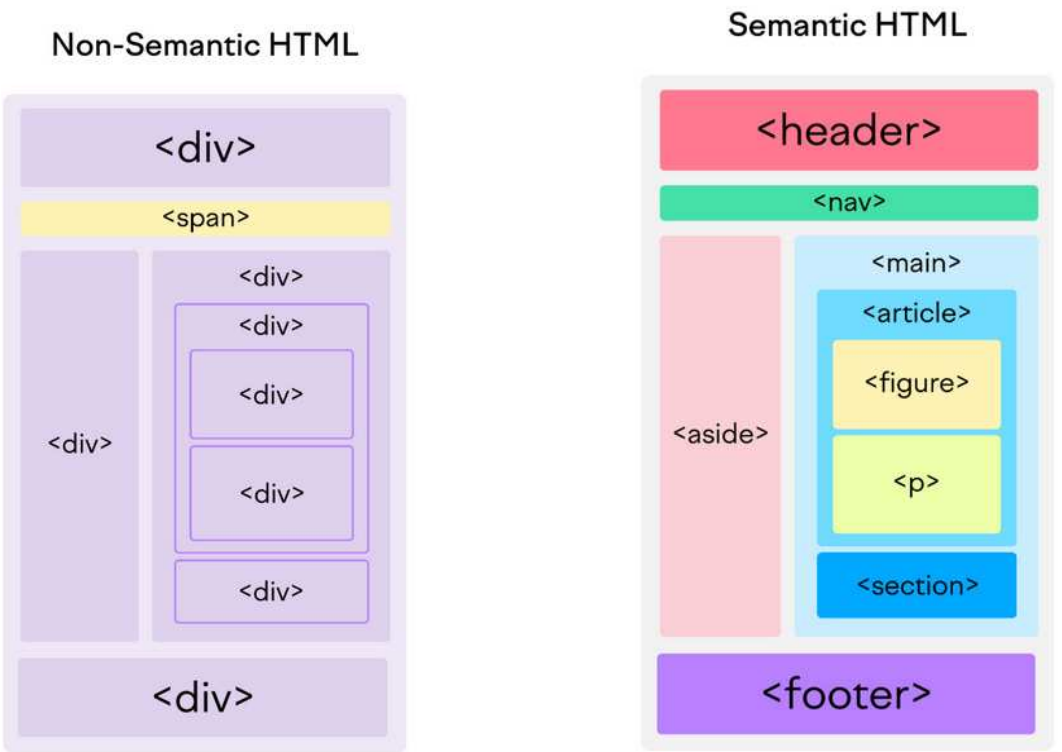
Output:



HTML Semantic Elements:

A semantic element clearly describes its meaning to both the browser and the developer and are example of non-semantic elements which tells nothing about of semantic elements which clearly defines its content. Following are some semantic elements used in HTML.

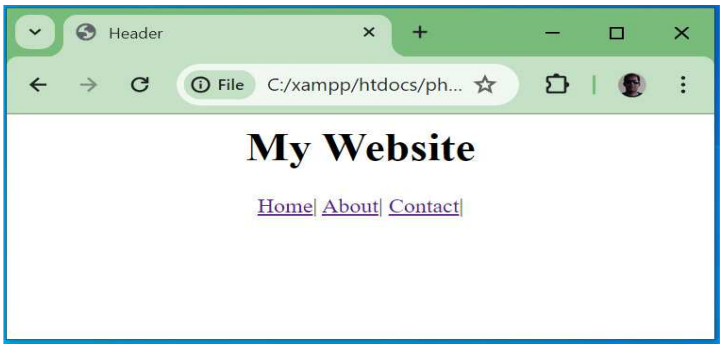
- **<header>:** Used to represent the top section of a web page, often containing headings, logos, and navigation.
- **<nav>:** Signifies a navigation menu on a web page.
- **<article>:** Indicates a self-contained piece of content, such as a blog post or news article.
- **<section>:** Represents a thematic grouping of content on a web page.
- **<aside>:** Typically used for sidebars or content that is tangentially related to the main content.
- **<footer>:** Represents the footer of a web page, usually containing copyright information and contact details.
- **<figure> and <figcaption>:** Used for embedding images, diagrams, or charts, along with a caption.
- **<main>:** Signifies the main content area of a web page.
- **<time>:** Used to represent time-related information, like dates and times.



Example of header and nav:

```
<html>
  <head>
    <title>Header </title>
  </head>
  <body>
        <header style="text-align: center;">
                <h1>My Website</h1>
                <nav>
                        <a href="#">Home</a>|
                        <a href="#">About</a>|
                        <a href="#">Contact</a>|

                </nav>
        </header>
  </body>
</html>
```

Output:

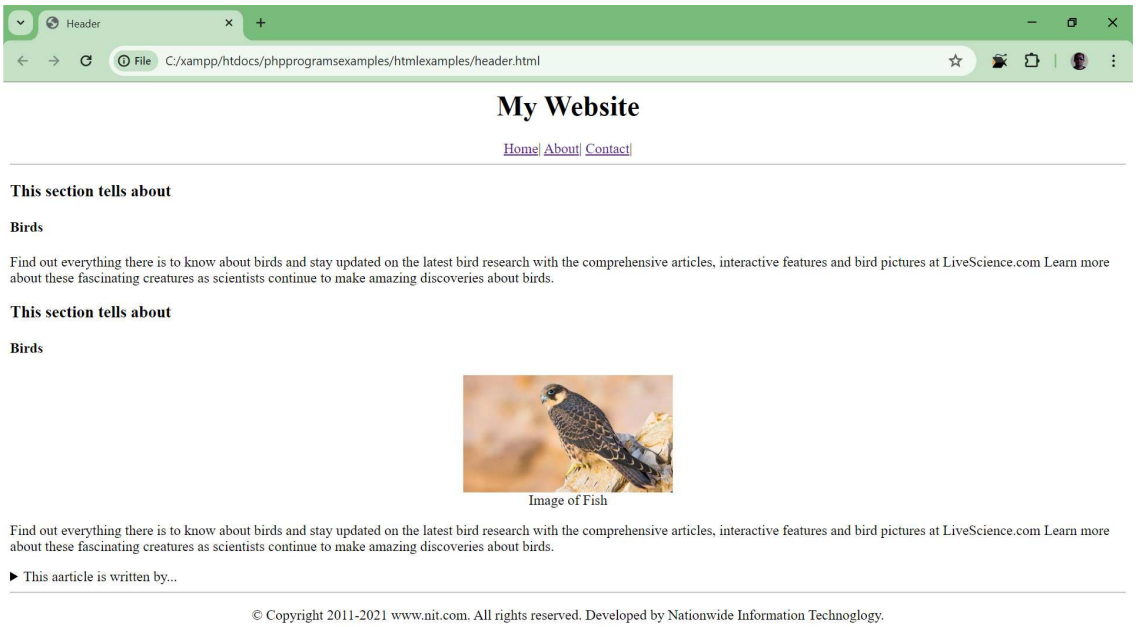


Example:

```
<html>
  <head>
    <title>Header </title>
  </head>
  <body>
    <header style="text-align: center;">
  <h1>My Website</h1>
  <nav>
    <a href="#">Home</a>|
    <a href="#">About</a>|
    <a href="#">Contact</a>|

  </nav>
</header>
<hr>
<section>
  <h3>This section tells about</h3>
    <article>
      <h4>Birds</h4>
      <p>Find out everything there is to know about birds and stay updated on the latest bird research with the comprehensive articles, interactive features and bird pictures at LiveScience.com Learn more about these fascinating creatures as scientists continue to make amazing discoveries about birds.</p>
    </article>
</section>
<section>
  <h3>This section tells about</h3>
    <article>
      <h4>Birds</h4>
      <figure style="text-align: center;">
```

```
            <img src="C:\\xampp\htdocs\phpprogramsexamples\htmlexamples\bird.webp" width="20%">
            <figcaption>Image of Fish</figcaption>

        </figure>
        <p>Find out everything there is to know about birds and stay updated on the latest bird research with the comprehensive articles, interactive features and bird pictures at LiveScience.com Learn more about these fascinating creatures as scientists continue to make amazing discoveries about birds.</p>
    </article>
</section>
<details>
    <summary>This aarticle is written by...</summary>
    <p>When we make the Guardian available to you online, we and our partners may use cookies and similar technologies to help us to do this. Some are necessary to help our website work properly and can't be switched off, and some are optional but support the Guardian and your experience in other ways. To do this we work with a cross section of 160 partners.</p>
    <p>Cookies and other similar technologies may be used to access personal data, including page visits and your IP address. We use this information about you, your devices and your online interactions with us to provide, analyse and improve our services. Depending on your choice, we may also use your data to personalise content or advertising.</p>
</details>
<footer style="text-align: center;">
    <hr>
    <p>© Copyright 2011-2021 www.nit.com. All rights reserved. Developed by Nationwide Information Technoglogy.</p>

</footer>
    </body>
</html>
```

Output:



HTML Events:

An event is a notification that is triggered when something changes in the browser. For example, when a user clicks a button, a click event occurs.

HTML Event Attribute Types

There are few categories for Event attributes and these are based on the event types supported in JavaScript. These are:

- Window Event Attributes
- Mouse Event Attributes
- Form Event Attributes
- Keyboard Event Attributes
- Media Event Attributes

➢ Window Event Attributes

Window Events are those events that are related to the window object, and these apply to the HTML `<body>` tag.

Attribute	Description
onafterprint	Executed the script after the document is printed.
onbeforeprint	Executed the script before the document is printed.
onbeforeunload	Executed the script before a document being unloaded.
onerror	Executed the script when an error occurs.
onhashchange	Executed the script when the anchor part in URL of the webpage is changed.
onload	Executed the script when the webpage is entirely loaded.
onmessage	Executed the script when a message event occurs.
onoffline	Executed the script when the network connection is disconnected, and browser started working offline.
ononline	Executed the script when the browser started working online
onpagehide	Executed the script when the current webpage is hidden such as if the user has moved away from the current webpage.
onpageshow	Executed the script when the current webpage is focused.
onpopstate	Executed the script when the window's active history is changed.
onresize	Executed the script when the window is resized.
onstorage	Executed the script when web storage is updated.
onunload	Executed the script when the current webpage is unloaded, or window is closed.

- **Mouse Event Attributes**

  Mouse events are those events that occur when the user interacts with the pointing device like a mouse.

The event attributes under the Mouse event category are given below:

Attributes	Description
onclick	This attribute fires at the time when the user clicks the left mouse button on the element.
ondblclick	This attribute fires at the time when the user double-clicks on the element.
oncontextmenu	This attribute fires at the time when a context menu is triggered by the user through the right-click on the element.
ondrag	This attribute fires when the user drags an element. The event ondrag got fires throughout the drag operation.
ondragend	This attribute fires when the user releases the mouse button at the end of a drag operation.
ondragenter	This attribute fires when the user drags an element to a valid drop target.
ondragleave	This attribute fires when an element leaves a valid drop target during a drag operation.
ondragover	This attribute fires when an element is being dragged over a valid drop target.
ondragstart	This attribute fires when the user starts to drag a text selection or selected element.
ondrop	This attribute fires at the time when the mouse button is released during a drag-and-drop operation i.e. when the dragged element is being dropped.
onmousedown	This attribute fires when the mouse button is pressed over an element.
onmousemove	This attribute fires when the user moves the mouse pointer over an element.
onmouseout	This attribute fires when the user moves the mouse pointer outside the boundaries of an element.
onmouseover	This attribute fires when the user moves the mouse pointer onto an element.
onmouseup	This attribute fires when the user releases the mouse button while the mouse is over an element.
onscroll	This attribute fires when the user scrolls the contents of an element by scrolling the element's scrollbar.
onshow	This attribute fires when a context menu was fired onto an element that also has a contextmenu attribute.
ontoggle	This attribute fires when the user opens or closes the <details> tag
onwheel	This attribute fires when the user scrolls the contents of an element by rolling the mouse wheel up or down over an element.

- **Form Event Attributes**

Form Events are those events in HTML, which occurs when any user **interacts** with the **form controls.**

The event attributes under the Form event category are given below:

Attributes	Description
onblur	This attribute fires when the element loses its focus.
onchange	This attribute fires when either the value or state of an element is changed
onfocus	This attribute fires when the element receives its focus.
oninput	This attribute fires at the time change in the value of an element by the user.
oninvalid	This attribute is fired at the time when a submittable element does not satisfy the constraints during form validation
onreset	This attribute fires at the time when the form is reset by the user.
onselect	This attribute fires when some part of the text is being selected or there is a change in current selection by the user.
onsearch	This attribute fires when the user does write some text in the searching field.
onsubmit	This attribute is fired at the time of form submission.

- **Keyboard Event Attributes**

Keyboard Events are those events that occur at the time when a user interacts with the keyboard.

The event attributes under the Keyboard event category are given below:

Attributes	Description
onkeydown	This attribute fires when a user presses any key on the keyboard.
onkeyup	This attribute fires at the time of alphanumeric key pressed by the user
onkeypress	This attribute fires when a user releases any key on the keyboard.

- **Media Event Attributes**

Media Events are those events that occur at the time of **handling events** of media elements that are embedded inside HTML Documents, for example `<audio>` tag, `<video>` tag, `<embed>` tag, etc.

The event attributes under the Media event category are given below:

Attributes	Description
onabort	This attribute fires when playback is aborted, but not due to an error.
oncanplay	This attribute fires at the time when enough data is available to play the media, at least for a couple of frames, but it would require further buffering.
oncanplaythrough	This attribute fires when the entire media can be played to the end without requiring to stop for further buffering.
oncuechange	This attribute fires when the text track cue of an <track> element changes
ondurationchange	This attribute fires when the duration of the media changes
onemptied	This attribute fires when the media element gets reset to its initial state, either because of a fatal error during the time of loading or because of the load() method is called to reload it.

onended	This attribute fires when the end of playback is reached
onerror	This attribute fires when there occurs an error at the time of fetching the data of media
onloadeddata	This attribute fires when media data is loaded at the current playback position
onloadedmetadata	This attribute fires when metadata of the media has finished loading. Duration and Dimensions are metadata of the media.
onloadstarts	This attribute fires when the loading of media starts.
onplay	This attribute fires when the playback of the media starts after having been paused i.e. when the play() method is requested.
onplaying	This attribute fires when the audio or video is playing
onpause	This attribute fires when the playback gets paused; it can be paused either programmatically or by the user.
onprogress	This attribute fires periodically which indicates the progress of downloading the media data
onratechange	This attribute fires when the playback rate or speed is increased or decreased, like slow motion or fast forward mode.
onseeking	This attribute fires when the current playback position is moved.
onseeked	This attribute fires when the seek operation comes to an end.
onstalled	This attribute fires at the time when the download has stopped unexpectedly.
onsuspend	This attribute fires at the time when the loading of the media is intentionally stopped.
ontimeupdate	This attribute fires when the playback position changed, like when the user fasts forwards to a different playback position.
onvolumechange	This attribute fires the time of change in volume like from muted to unmuted
onwaiting	This attribute fires when playback stops because the next frame of a video resource is not available.