UNIT-2

Hyper Text Markup Language

writing code for given tables & forms is imp in this unit.
OR creating those writing html code.

HTML stands for Hyper Text Markup Language. It is the standard markup language for web pages. HTML consists of a series of elements and tells the browser how to display the content. HTML elements are represented by tags. Browsers do not display the HTML tags, but use them to render the content of the page. With the help of HTML we can create the structure of our website. HTML is like skeleton of human body for any website, which gives base to the structure.

⊛ Structure of HTML with example: [Imp]

This is example & explanation

```
<!DOCTYPE html>
<html>
    <head>
        <title> Page Title </title>
    </head>

    <body>
        <h1> This is a heading. </h1>
        <p> This is a paragraph. </p>
    </body>
</html>
```

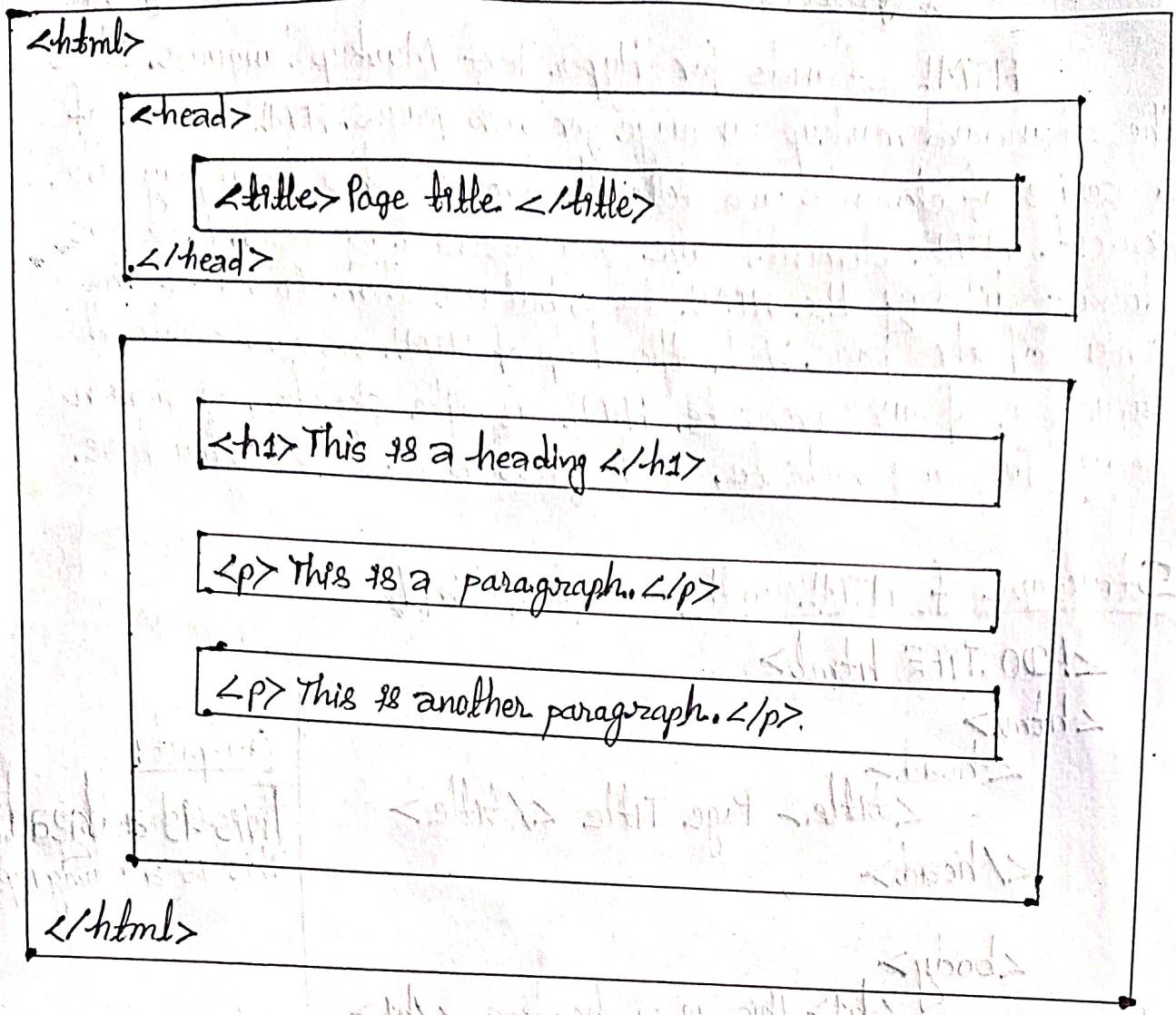
Output:

This is a heading.
This is a paragraph.

Note:-
- <...> → denotes opening tag.
- </...> → denotes closing tag

- → The <!DOCTYPE html> declaration defines this document to be HTML5.
- → html, head, body, title, h1, p etc. are html elements. The symbol < > denotes opening tag of an element and symbol </ > denotes closing tag of an element.
- → The <html> element is the root element of an HTML page.
- → The <head> element contains <title> element which describes or names the title of document and it also contains meta tags.
- → The <body> element contains the visible page content.
- → The <h1> element defines a large heading.
- → The <p> element defines a paragraph.

HTML Page Structure



⊛. HTML Elements:

HTML element is defined by a start tag, some content, and an end tag. The HTML element is everything from the start tag to the end tag:

<tagname> Content goes here... </tagname>

Examples:

<h1> My First Heading </h1>
 My first paragraph.

Here,

Start tag	Element Content	End tag
<h1>	My First Heading	</h1>
	My first paragraph.	
 	none	none

Note: Some HTML elements have no content (like
 element) which provides some space. These are empty elements and do not have end tag.

⊛. HTML Attributes:

All HTML elements can have attributes. Attributes provide additional information about elements. Attributes are always specified in the start tag. Attributes usually come in name/value pairs like: name="value". Following are some of the html attributes:

i) **The href attribute:** The <a> tag defines a hyperlink. The href attribute specifies the URL of the page the link goes to:
Example: Visit NoteJunction

ii) **The src attribute:** The <img> tag is used to embed an image in an HTML page. The src attribute specifies the path to the image to be displayed:
Example: <img src="image_girl.jpg">

iii) **The width and height attributes:** The <img> tag also contain the width and height attributes, which specifies the width and height of the image (in pixels).
Example: <img src="image_girl.jpg" width="500" height="600">

iv) **The alt attribute:** The alt attribute for the <img> tag specifies an alternate text for an image, if the image for some reason cannot be displayed. This can be due to slow connection, or an error in the src attribute.
Example:- <img src="image_girl.jpg" alt="Girl with a jacket">

⊛. HTML Headings:

HTML headings are titles or subtitles that we want to display on a webpage. HTML headings are defined with the <h1> to <h6> tags. <h1> defines the most important heading and has the largest font size while <h6> defines the least important heading and has the smallest font size. Browsers automatically add some white space (margin) before and after a heading. It is important to use headings to show the document structure. <h1> headings should be used for main headings, <h2> for sub-headings and so on as our requirement.

Example:

```
<!DOCTYPE html>
<html>
  <body>
    <h1>This is heading 1</h1>
    <h2>This is heading 2</h2>
    <h3>This is heading 3</h3>
    <h4>This is heading 4</h4>
    <h5>This is heading 5</h5>
    <h6>This is heading 6</h6>
  </body>
</html>
```

Output:

This is heading 1
This is heading 2
This is heading 3
This is heading 4
This is heading 5
This is heading 6

⊛. HTML Paragraphs:

The paragraph always starts on a new line, and is usually a block of text. The html <p> element defines a paragraph. It also adds some white space (margin) before and after a paragraph.

Example:

```
<!DOCTYPE html>
<html>
<body>
    <p>This is a paragraph.</p>
    <p>This is another paragraph.</p>
</body>
</html>
```

Output:

This is a paragraph.
This is another paragraph.

⊛. HTML Division:

The tag defines a division or a section in an HTML document. The tag is used as a container for HTML elements which is then styled with CSS or manipulated with JavaScript. The tag is easily styled by using the class or id attribute. Any sort of content can be put inside the tag like html headings, html paragraphs, html forms, html tables etc. By default, browsers always place a line break before and after the element.

⊛. HTML Comments:

HTML comments are not displayed in the browser, but they can help document our HTML source code. We can add comments to our HTML source by using following syntax:

<!-- Write comments here -->

Comments can be used to hide content which may help us to write things useful to understand code in better way but it should be hidden from document or it should not be executed. We can hide one line or more than one line, everything between the <!-- and the -->.

⊛ Formatting:

HTML formatting is used for formatting text. Formatting elements were designed to display special types of text.

 → Bold text
 → Important text
 → Italic text
 → Emphasized text
<mark> → Marked text
<small> → Smaller text
<del> → Deleted text
<ins> → Inserted text
<sub> → Subscript text
<sup> → Superscript text.

1) HTML and elements:

The HTML element defines bold text, without any extra importance.

Example: This text is bold.

The HTML element defines text with strong importance. The content inside is typically displayed in bold.

Example: This text is important!.

2) HTML and elements:

The HTML element defines a part of text in an alternate mood. The content inside is typically displayed in italic.

Example: This text is italic

The HTML element defines emphasized text. The content inside is typically displayed in italic. Example: This text is emphasized..

iii) HTML <small> element:

The HTML <small> element defines smaller text.

Example:- <small>This is some smaller text.</small>.

iv) HTML <sub> element:

The HTML <sub> element defines subscripted text. Subscript text appears half a character below the normal line and is sometimes rendered in a smaller font. Subscript text can be used for displaying chemical formulas, like H_2O.

Example: H ₂ O

v) HTML <sup> element:

The HTML <sup> element defines superscript text. Superscript text appears half a character above the normal line, and is sometimes rendered in a smaller font. Superscript text can be used for footnotes, like WWW[1].

Example: WWW ¹ .

Spacing:

Spacing is used to maintain space between texts. We have several options for creating and controlling white space on our website following are some of them;

i) HTML
 element:

The HTML
 tag denotes a line break. We use this tag if we want to display upcomming text in the next line.

ii) HTML <pre> element:

The HTML <pre> tag is used with preformatted text. It instructs the browser that the text is to appear exactly as written in the HTML file, including any spaces or blank lines. If we write five spaces inside <pre> tags, then we get five spaces on the website.

iii)   → This will add single white space.

iv)   → This will add two white spaces.

v)   → This will add four white spaces.

Formatting text phrases

1) The element: The tag is an inline container used to mark up a part of a text, or a part of a document. The tag is much like element, but is a block-level element and is an inline element. The tag supports global and event attributes.

2) The <tt> element: The <tt> tag was used in HTML 4 to define teletype text.

⊛. Image Element:

The HTML <img> tag is used to embed an image in a web page. Images are not technically inserted into a web page rather images are linked to web pages. The <img> tag creates a holding space for the refrenced image. The <img> tag is empty, it contains attributes only, and does not have a closing tag. The <img> tag has two required attributes:

src → specifies the path to the image
alt → specifies the alternate text for image.

Syntax:

```
<img src="url" alt="alternate text">
```

Example:

```
<img src="img_roshan.jpg" alt="Csit Student">
```

When a web page loads, the browser gets the image from server or local file and inserts into the page. The src attribute should be given correct path or url, otherwise we will get a broken link icon on the browser, along with alt text value.

If the user for some reason cannot view image because of slow connection or an error in the src attribute or any other reason then the value of alt attribute describes the image. We can use style attribute to specify width and height of image.

Example:

```
<img src="img_roshan.jpg" alt="csit student" style="width:500px;
height:600px;">
```

Images in another folder:

If we have images in a sub-folder, we must include folder name in the src attribute as in the example below:

Example: `<img src="/images/img_roshan.jpg" alt="student" style="width:320px; height:480px;">`

Images on another server/website:

Some websites point or display an image that is on another server/website. To use or point an image on another server, we must specify full url in the src attribute:

Example: `<img src="https://www.notejunction.com/images/img.jpg" alt="Note Junction">`

Background Image:

A background image can be specified for almost any HTML element. To add a background image on an HTML element, use the HTML style attribute and the CSS background-image property:

Example: `<div style="background-image: url('img_roshan.jpg');">`

Anchor tag (<a> tag):

An Anchor tag in HTML can be defined as a means to create a hyperlink that can link our current page on which the text is being converted to hypertext via <a> (anchor tag) to another page. This anchoring from one page to another is made possible by the attribute "href".

The attribute "href" of the anchor tag is implemented for defining the address or path to which this hypertext will get linked. The syntax for an anchor tag with href attribute looks something like this:

`<a href="URL">Text Here</a>`

→ An unvisited link gets displayed with properties like underlined and with the color blue.

→ A link that is visited gets displayed as underlined with color as purple.

→ A link that is an active link gets displayed as underlined with red color.

⊛. Lists:

HTML lists allow web developers to group a set of related items in lists.

i) Unordered list: An unordered list starts with the <ul> tag. Each list item starts with the <li> tag. The list items will be marked with bullets (small black circles) by default:

Example:
```
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

Output:
- Coffee
- Tea
- Milk

ii) Ordered list: An ordered list starts with the <ol> tag. Each list item starts with the <li> tag. The list items will be marked with numbers by default:

Example:
```
<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

Output:
1. Coffee
2. Tea
3. Milk

Note:

i) Unordered list items are marked with circles by default but we can change it specifying style attribute and list-style-type property with values like circle, square, none etc. in the <ul> tag.

Example:
```
<ul style="list-style-type:square">
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

Output:
- Tea
- Milk

ii) Similarly ordered list are marked with numbers but we can change it specifying type attribute in <ol> tag.

Example:
```
<ol type="i">
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

Output:
i. Tea
ii. Milk

⊛. Tables: [Imp]

HTML table allows web developers to arrange data into rows and columns. It is defined with the <table> tag. ~~Each~~ A table header is defined with the <th> tag. Each table row is defined with the <tr> tag. By default, table headings are bold and centered. A table data/cell is defined with the <td> tag.

Example:

```
<table>
    <tr>
        <th>Company</th>
        <th>Contact</th>
        <th>Country</th>
    </tr>

    <tr>
        <td>A1 Soft</td>
        <td>Sandeep Singh</td>
        <td>Nepal</td>
    </tr>

    <tr>
        <td>Island Trading</td>
        <td>Helen Bennet</td>
        <td>UK</td>
    </tr>
</table>
```

Output:

Company	Contact	Country
A1 Soft	Sandeep Singh	Nepal
Island Trading	Helen Bennet	UK

Company Contact Country
A1 Soft Sandeep Singh Nepal
Island Trading Helen Bennet UK

A border is set using the CSS border property. If we do not specify a border for the table, it will be displayed without borders. Let we use css property in the above example as below:

```
table, th, td {
    border: 1px solid black;
    border-collapse: collapse;
}
```

Now the Output with css will be as follows:

Company	Contact	Country
A1 Soft	Sandeep Singh	Nepal
Island Trading	Helen Bennet	UK

Cell Padding → cellpadding attribute can be specified in the <table> to get space between the cell content and it's borders.

Example:-
```
<table cellpadding="5px">
    :
</table>
```

Align → Table headings and table data can be ~~alig~~ aligned to left, right, center etc using css property text-align. Let we have to align table headings to center then it can be done as;

```
th {
    text-align: center;
}
```

Column Span (colspan): To make a cell span more than one column, we use the colspan attribute.

Example:
```
<table style="width:100%">
    <tr>
        <th>Name</th>
        <th colspan="2">
            Telephone
        </th>
    </tr>

    <tr>
        <td>Bill Gates</td>
        <td>55578546</td>
        <td>57584567</td>
    </tr>
</table>
```

Output:

Name	Telephone	
Bill Gates	55578546	57584567

Row span (rowspan): To make a cell span one or more than one row, we use ~~rowspan~~ rowspan attribute.

Example:

```
<table style="width:100%">
  <tr>
    <th>Name</th>
    <td>Bill Gates</td>
  </tr>
  <tr>
    <th rowspan="2">Telephone</th>
    <td>55577854</td>
  </tr>
  <tr>
    <td>55577855</td>
  </tr>
</table>
```

looks similar to colspan but not the th and tr are aggarranged in a way so that it looks like proper table

Output:

Name	Bill Gates
Telephone	55577854
	5577855

Table Caption: We can add caption that serves as a heading for the entire table. To add caption to a table, we use the <caption> tag immediately after the <table> ~~tag~~ tag.

Example:

```
<table style="width:100%">
  <caption>Monthly Savings</caption>
  <tr>
    <th>Month</th>
    <th>Savings</th>
  </tr>
  <tr>
    <td>January</td>
    <td>$100</td>
  </tr>
  <tr>
    <td>February</td>
    <td>$50</td>
  </tr>
</table>
```

Output:

Monthly Savings

Month	Savings
January	$100
February	$50

⊛. HTML Frames (iframe): The "iframe" in html stands for inline frame. The "iframe" tag defines a rectangular region within the document in which the browser can display a separate document, including scrollbars and borders. The 'src' attribute is used to specify the URL of the document that occupies the iframe.

Syntax: <iframe src="URL" title="description"></iframe>

Many times we see youtube videos are embedded within newsportal and other websites. This is done with the help of iframe. The embedded link of youtube video is place in the src of iframe to make this all happen.

⊛. Forms: [Imp]

HTML form is used to collect user input. The user input is most often sent to a server for processing. The HTML <form> element is used to create an HTML form for user input. The <form> element is a container for different types of input elements such as; text, checkboxes, radio, submit buttons etc.

1) The <input> Element: The HTML <input> element is most used form element. An <input> element can be displayed in many ways, depending on type attribute. Here are some examples:

Type	Description
<input type="text">	Displays a single-line text input field.
<input type="radio">	Displays a radio button (for selecting one of many choices).
<input type="checkbox">	Displays a checkbox (for selecting zero or more of many choices).
<input type="submit">	Displays a submit button.
<input type="button">	Displays a clickable button.

2) The <label> Element: The <label> tag defines a label for many form elements. The <label> element is useful for screen-reader users. to distinguish between different fields like username, password etc.

Example: A form with radio buttons:

```
<!DOCTYPE html>
<html>
<body>
    <h2>Radio Buttons</h2>
    <p>Choose your favourite web language:</p>

    <form>
      <input type="radio" id="html" name="fav_language" value="HTML">
      <label for="html">HTML</label> <br>
      <input type="radio" id="css" name="fav_language" value="CSS">
      <label for="css">CSS</label> <br>
      <input type="radio" id="javascript" name="fav_language" value="JavaScript">
      <label for="javascript">Java Script</label>
    </form>
</body>
</html>
```

Output:

Radio Buttons

Choose your favourite web language:

- o HTML
- o CSS
- o Java Script.

⊛. HTML <form> Elements:

The HTML <form> element can contain one or more of the following form elements:

- <input>
- <label>
- <select>
- <textarea>
- <button>
- <fieldset>
- <legend>
- <datalist>
- <output>
- <option>
- <optgroup>

3) The <select> Element:

The <select> element defines a drop-down list.

Example:

```
<form>
    <label for="cars"> Choose a car: </label>
    <select>
        <option value="volvo"> Volvo </option>
        <option value="saab"> Saab </option>
        <option value="fiat"> Fiat </option>
        <option value="audi"> Audi </option>
    </select>
    <input type="submit">
</form>
```

4) The <option> Element:

The <option> elements defines an option that can be selected. By default, the first item in the drop-down list is selected. To define a pre-selected option, add the ~~selected~~ selected attribute to the option.

Example: `<option value="audi" selected> Fiat </option>`

We use the ~~size~~ attribute to specify the number of visible values.

Example:

```
<select size="3">
    <option> ... </option>
    <option> ... </option>
    :
    <option> ... </option>
</select>
```

Similarly we use multiple attribute to allow the user to select more than one value.

5) The <textarea> Element:

The <textarea> element defines a multi-line input field. The rows attribute specifies the visible number of lines in a text area. The cols attribute specifies the visible width of a text area.

Example:

```
<textarea name="message" rows="10" cols="30">
    The cat was playing in the garden.
</textarea>
```

⊛. Form Attributes:

i) The action attribute: The <action> attribute defines the action to be performed when the form is submitted. Usually, the form data is sent to a file on the server when the user clicks on the submit button. In the example below, the form data is sent to a file called "action_page.php". This file contains a server-side script that handles the form data:

Example:
```
<form action="/action_page.php">
  : <!--form elements -->
</form>
```

#If the <action> attribute is omitted, the action is set to the current page.

ii) The target attribute: The target attribute specifies where to display the response that is received after submitting the form. The target attribute has default value _self which means that the response will open in the current window. The target attribute can have one of the following values:

Value	Description
_blank	The response is displayed in new tab.
_self	The response is displayed in current tab.
_parent	The response is displayed in parent frame.
_top	The response is displayed in full body of window or tab.

Example: Here, the submitted result will open in a new browser tab:

```
<form action="/action_page.php" target="_blank">
```

iii) The method attribute: The method attribute specifies the HTTP method to be used when submitting the form data. The form-data can be sent as URL variables (with method="get") or as HTTP post transaction (with method="post"). The default HTTP method when submitting form data is GET.

Example:
```
<form action="/action_page.php" method="get">
```

iv) The autocomplete Attribute:

The autocomplete attribute specifies whether a form should have autocomplete on or off. When autocomplete is on, the browser automatically complete values based on values that the user has entered before.

Example: <form action="/action_page.php" autocomplete="on">

v) The novalidate Attribute:

The novaldidate attribute is a boolean attribute. When present, it specifies that the form-data (input) should not be validated when submitted.

Example: <form action="/action_page.php" novalidate>

HTML Input Types:

<input type="button">
<input type="checkbox">
<input type="color">
<input type="date">
<input type="datetime-local">
<input type="email">
<input type="file">
<input type="image">
<input type="number">
<input type="password">
<input type="radio">
<input type="range">
<input type="reset">
<input type="search">
<input type="submit">
<input type="tel">
<input type="text">
<input type="url">
<input type="week">

HTML Input ~~value~~ Attributes:

i) value attribute:- The input value attribute specifies an intial value for an input field.

Example: <input type="text" value="John">

ii) readonly attribute:- The input readonly attribute specifies that an input field is read-only it can't be modified.

Example: <input type="text" value="John" readonly>

iii) disabled attribute:- The input disabled attribute specifies that an input field should be disabled. A disabled field is unusable and un-clickable. The value of a disabled input field will not be sent when submitting the form.

Example: <input type="text" value="John" disabled>

iv) size attribute:- The input size attribute specifies the visible width, in characters, of an input field. The default value for size is 20.

Example:- <input type="text" name="pin" size="4">

v) The min and max attributes:- The input min and max attributes specify the minimum and maximum values for an input field.

Example: <input type="number" min="1" max="5">

vi) placeholder attribute: The input place-holder attribute specifies a short hint that describes the expected value of an input field.

vii) required attribute: The input required attribute specifies that an input field must be filled out before submitting the form.

[Imp] ⊛ Based on form knowledge try to practice login and signup forms. In exam theory is not asked, in many cases creating login and signup forms or other forms are asked. Go through youtube tutorials it's easy. [Theory is for base knowledge only].

⊛ Meta Tag:

The <meta> tag defines metadata about an HTML document. Metadata is information about data. <meta> tags always go inside the <head> element, and are typically used to specify character set, page description, keywords, author of the document, and viewport settings. Metadata will not be displayed on the ~~paged~~ page, but is machine parsable. Metadata is used by browsers, search engines and other web services.

Examples:- i) Define keywords for search engines:

<meta name="keywords" content="HTML, CSS">

ii) Refresh document in every 30 seconds.

<meta http-equiv="refresh" content="30">

iii) Setting the viewport to make website look good on all devices:

<meta name="viewport" content="width=device-width, initial-scale=1.0">

⊛. HTML Audio:

The HTML <audio> element is used to play an audio file on web page.

Example:-

```
<audio controls>
    <source src="horse.ogg" type="audio/ogg">
    <source src="horse.mp3" type="audio/mpeg">
</audio>
```

The controls attribute adds audio controls like play, pause and volume. The <source> element allows us to specify path or url to audio we are going to use. The browser will use the first recognized format. To start an audio file automatically we use autoplay attribute. Similarly we can use muted attribute to mute the audio. The preload attribute specifies how the audio should be loaded when the page loads.

Preload Syntax: <audio preload="auto|metadata|none">

where, auto → Entire audio file is loaded when page loads.
metadata → The browser should load only metadata when page loads.
none → The browser should not load audio file when page loads.

The loop attribute is a boolean attribute. When present, it specifies that the audio will start over again, every time it is finished.

⊛. HTML Video:

The HTML <video> element is used to show a video on web page.

Example:

```
<video width="320" height="240" controls>
    <source src="movie.mp4" type="video/mp4">
</video>
```

The controls attribute adds video controls like play, pause and volume. It is good idea to always include width and height attributes. If height and width are not set, the page might flicker while the video loads. The <source> element allows us to specify path or url to video we are going to use. To start video automatically, we use autoplay attribute. The loop attribute is a boolean attribute. When present, it specifies that the video will start over again, every time it is finished. The poster attribute is used to display the image while video downloading or when user click the play button. If this image is not set then it will take first frame of video as poster image. Syntax: <video poster="URL">.

⊛. Canvas:

The HTML <canvas> element is used to draw graphics on a web page. The <canvas> element is only a container for graphics. We must use Javascript to actually draw the graphics. Canvas has several methods for drawing paths, boxes, circles, text and adding images. A canvas is a rectangular area on an HTML page. By default, a canvas has no border and no content.

Example: Drawing a line in canvas:

```
<canvas id="myCanvas" width="200" height="100"></canvas>
```

Adding JavaScript

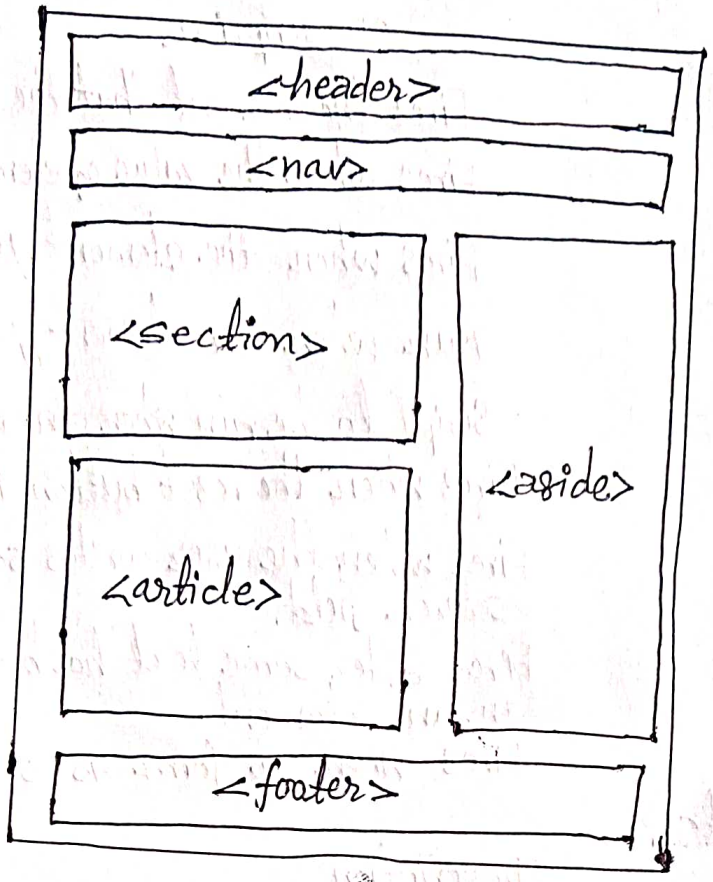
```
<script>
    var c=document.getElementById("myCanvas");
    var ctx=c.getContext("2d");
    ctx.moveTo(0,0);
    ctx.lineTo(200,100);
    ctx.stroke();
</script>
```

for circle we will change these two lines as:
ctx.beginPath();
ctx.arc(95,50,40,0,2*Math.PI);
others all same

⊛. HTML Semantic Elements:

A semantic element clearly describes its meaning to both the browser and the developer. and are examples of non-semantic elements which tells nothing about its content. <form>, <table> and <article> are some examples of semantic elements which clearly defines its content. Following are some semantic elements used in HTML:

i) <article> → defines independent, self-contained content.
ii) <aside> → defines content aside from page content.
iii) <details> → defines additional details that user can view or hide.
iv) <footer> → defines a footer for a document.
v) <header> → defines a header for a document.
vi) <main> → specifies the main content of a document.
vii) <nav> → defines navigation links
viii) <section> → defines a section in document.



Fires meaning runs

HTML Events:

1) Window Events: HTML has the ability to let events trigger actions in a browser, like starting a Javascript when a user clicks on an element. Events triggered for the window object applies to the <body> tag:

Attribute	Description
onerror	Script to be run when error occurs
onload	Script to be ~~rload~~ run after page finishes loading.
onmessage	Script to be run when message is triggered.
onoffline	Script to be run when browser starts to work offline.
ononline	Script to be run when browser starts to work online.
onresize	Fires when the browser window is resized.
onafterprint	Script to be run after document is printed.
onbeforeprint	Script to be run before document is printed.

2) Form Events: Events triggered by actions inside a HTML form (applies to almost all HTML elements, but is most used in form elements):

Attribute	Description
onblur	Fires the moment that the element loses focus.
onchange	Fires when the value of element is changed.
onfocus	Fires when the element gets focus.
oninput	Run when an element gets user input
oninvalid	Script to be run when an element is invalid.
onreset	Fires when the reset button in form is clicked.
onsearch	Fires when the user writes something on search field.
onselect	Fires after some text has been selected in an element.
onsubmit	Fires when a form is submitted.

3) Keyboard Events:

Attribute	Description
onkeydown	Fires/runs when a user pressing a key.
onkeypress	Fires when user presses key.
onkeyup	Fires when user releases a key.

4) Mouse Events:

Attribute	Description
onclick	Fires on a mouse click on the element.
ondblclick	Fires on a mouse double-click on the element.
onmousedown	Fires when a mouse is pressed down on an element.
onmouseout	Fires when mouse pointer moves out of an element.
onmouseover	Fires when mouse pointer moves over an element.
onmouseup	Fires when mouse button is released over an element.
~~onmousewheel~~	
onmousemove	Fires when the mouse pointer is moving while it is over an element.
onwheel	Fires when the mouse wheel rolls up or down over an element.