Assignment No :- 1

[unit : 1]

Describe the basic structure of an HTML document The basic structure of an HTML (Hyper text markup Language document consists of the following element.

1. Doctype Declaration

<! DOCTYPE html>

Declares the document type and version.

2. HTML Element

<html>
</html>

Root element of the HTML document.

3. Head Element

<head>
</head>

contains metadata about the document.

SILVER OAK UNIVERSITY
EDUCATION TO INNOVATION
NAVKAR ENTERPRISE
PAGE NO.
Date :

4. Title element

<title>
</title>

specifies the document title

5. Body element

<body>
</body>

contains the document's content.

Basic structure :-

<!DOCTYPE html>
<html>
<head>
<title> Document title </title>
</head>
<body>
<!... content goes here -->
</body>
</html>

Explanation:

1. <!DOCTYPE html> declares the document type.

2. <html> is the root element.
3. <head> contains metadata.
4. <title> specifies the document title.
5. <body> contains the document's content.

Optional Elements:

- <meta> : Provides additional metadata.
- <link> : links to external stylesheets or scripts.
- <script> : embeds Javascript code
- <style> : Defines internal CSS styles.

Best Practies:-

1. use lowercase tags and attribute names.

2. close tags in the reverse order they were opened.

3. use quotes around attribute values.

4. validate your HTML code.

Q.2 what are HTML Form elements and attributes? Provide examples of commonly used Form elements and their attributes.

Answer: Form Elements:

1. Input
- Type (text, password, email, number, checkbox,

radio, etc.

- name
- placeholder
- value
- Required

Example : <input type = "text" name = "username" Placeholder = " Enter username" required >

1. Textarea.

- Rows
- cols
- Name
- Placeholder

Example : < textarea rows = "5" cols = " 30" name = "description" Placeholder = " enter description" > < /textarea >

I. select

- Name
- multiple (optional)
- size (optional)

Example : < select name = " Color" multiple size = "3" > < option value = "red" > Red /option > < option value = "green" > Green < /option> < option value = " blue" > Blue < /option > < /select >

1. option
- value
- selected (optional)

Example: <option value = "red"
slected > Red </ option>

1. Button
- type (submit, reset, button)
- Name
- Value

Example: < button
type = "submit" name = "submit"
value = "submit" > submit </button >

1. label
- for (associates label with input)

Example: <label>
for = "username" > username : </lable >

1. Form

- Action (URL to submit Form data)
- method (GET, POST, etc)
- Enctype (optional)

Example :- < Form action = "/submit"
method = "Post" enctype = "multipart/
Form-data" > <!--- Form elements here ---> </Form>

Attributes :-

1. Required :- Specifies that the field must be field must be filled.

2. Placeholder :- Provides a hint to the user

3. Disabled :- Disables the field.

4. Readonly :- Prevents user input.

5. maxlength :- sets maximum input length,

6. min and max :- sets minimum and maximum values for numeric inputs.

7. Pattern : Specifies a regular Expression Pattern for input validation,

Example Form :-

< form action = "/submit"
method = "Post" >
</label for = "username" > username : </label>
<input type = "text"
name = "username" placeholder = " enter
username " required >

```
<br>
<label for = " Password " > Password : <l label>
<input type = " Password "
name = " Password " required >
<br>
<select name = " color " >
<option value = " red " > Red <l option>
<option value = " green " > Green <l option>
<option value = " blue " > Blue <l option>
<l select>
<br>
<button type = " submit "
name = " submit "
name = " submit " > submit <l button>
<l form>
```

This Example demonstrates various form elements and attributes.

Q.3 Discuss the importance of browser support in HTML development. How can developers ensure cross-browser compatibility?

Answer: Importance of Browser Support:

1. Reach wider audience: support multiple browser to cater to diverse user base.

2. Consistent user Experience :- Ensure uniform layout, functionality, and performance.

3. enhanced credibility :- cross - browser compatibility builds trust and professionalism.

4. improved accessibility : Ensure equal acess for user with disabilities.

5. Reduced maintenance : Test and fix issues once, rather than repeatedly.

Challenges in cross - Browser compatibility :

1. Browser rendering differences
2. CSS and Javascript implementation variations.
3. HTML 5 feature support disparities.
4. Device and screen size diversity.
5. operating system inconsistencies.

Ensuring cross - Browser compatibility :

Pre - Development :

1. Define target browsers and devices.

2. Research browser market share and trends.

3. set up testing environments.

During Development :

1. use HTML 5 and CSS 3 standards.

2. validate code using w3c validators

3. Test with browser - specific tools (e.g. chrome DEV Tools)

4. utilize cross - browser testing Frameworks (e.g. selenium.)

5. Implement responsive design.

Post - Development :-

1. test on multiple browsers and devices
2. conduct user testing and Feedback
3. monitor browser updates and adapt
4. continuously iterate and refine.

tools For cross - Browser testing

1. Browserstack
2. CrossBrowser testing
3. sauce labs
4. Selenium
5. VirtualBox

Best Practices :

1. separate ~~s~~ css and Javascript files
2. use CSS Prefixes (- webkit - moz. ms)
3. Avoid browser specific hacks.

4. use feature detection (e.g. modernizr)
5. Document browser - specific issues

Front end frameworks for cross Browser compatibility :

1. Bootstrap
2. Foundation
3. Material-ui
4. React
5. Angular

By prioritizing cross-browser compatibility, developers ensure a seamless user experience, enhance credibility, and reduce

Q.4 How can background colours and images be added to an HTML webpage? Provide code examples demonstrating both techniques?

Answer: Background colors and images can be added to an HTML ~~web~~ webpage using CSS (cascading style sheets) :

method 1 : inline CSS

Add background color or image directly to an HTML element using the style attribute.

Background color :

<div style = " back ground - color ;

```
url ('image.jpg') ; height:
500px ;" > content </div>
```

method 2: internal CSS

Define styles in the <head> section of the HTML document.

```
<head>
  <style>
  body {
    background color : #f2f2f2 ;
  }
  • background - image {
    background - image :
    url ("image.jpg');
    height : 500px
  }
  </style>
  </head>
  <body>
  <div
  class = " background -image" > content </div>
  </body>
```

method 3: External CSS

link an External CSS File to the HTML document.

Style.CSS:

```
body {
  background-color : #f2f2f2;
}
```

• background-image {
background-image:
url('image.jpg');
height: 500px;
}

index.html:

```
<head>
<link rel="stylesheet"
type="text/css" href="style.css">
</head>

<body>
<div
class="background-image"> content </div>
</body>
```

Background image Properties:

• background-repeat : repeat, no-repeat, repeat-x repeat-y

• background-position : center, top, bottom, left right.

Example :-

```
• background - image {
  background - image :
  url ('image.jpg') :
  background - repeat : no - repeat ;
  background - position : center ;
  background - size : cover ;
}
```

Responsive Background image :-

```
• background - image {
  background - image ;
  url ('image , Ipg');
  background - size : 100% 100%
  background - repeat : no - repeat ;
  heigh : 100vh ;
```

These methods demonstrate how to add background colors and images to an html webpage using CSS

Q.5 Explain div and span tag in details.

Answer:- div and span are two essential HTML elements used for structuring and styling content.

Div Tag :-

div stands for "division" or "container". It's a

block-level element used to group elements together for:

1. layout purposes
2. styling
3. semantic meaning

characteristics :-

1. Block-level element (occupies full with)
2. Can contain other elements (including text, images, etc.)
3. can be styled with CSS.
4. can have attributes (e.g. id, class, style)

Example :-

```
<div class = " Container">
  <h2> Heading </h2>
  <P> paragraph text </P>
  <img src ="image.jpg " alt = "Image">
</div>
```

Span Tag :-

Span is an inline element used to :

1. Apply styles to text.
2. Provide sematic meaning
3. Group text or inline elements.

characteristics :-

1. Inline element (only occupies space needed)
2. can contain text or other inline elements.
3. can be styled with CSS
4. can have attributes (e.g. id, class, style)

Example :-

```
<P> This is a <span
class="highlight"> important </span>
announcement. </P>
```

Key differences :-

1. Display type : div is block-level, while span is inline.

2. Purpose : div is for layout and containerization, while span is for text-level styling and semanties.

when to use each :-

1. use div for :
- Creating layout containers
- Grouping elements for styling
- Creating sections or divisions

2. use span for :
- Grouping inline elements
- Applying styles to text.

Best Practices :-

1. use semantic HTML elements (e.g. header, footer, nav) instead of div (when possible).

2. use span for text level styling, rather than div.

3. Avoid using div or span unnecessarily; use other semantic elements instead.

By understanding the differences and uses of div and span you can create more structured, accessible and maintainable HTML documents.

Assignment No :- 02
[unit :- 02]

Q.2 Explain the difference between headings (h1 - h6) and Paragraphs (<P> tag) in HTML

Answer Headings (h1 - h6)

Heading are used to define the title or heading of a section or page. There are six levels of headings, each representing a different level of importance:

1. h1 : most important, typically the main title of the page
2. h2 : secondary title, subdividing the content.
3. h3 : Tertiary title, further subdividing the content.
4. h4, h5, h6 : less important, used for subheadings.

Characteristics :-

1. Displayed in a larger font size and bold weight.
2. Typically used for titles, headings and subheadings
3. Search engines use headings to understand page structure.
4. Screen readers announce headings to users

Example :-

<h1> main title </h1>
<h2> secondary title </h2>
<h3> tertiary title </h3>

Paragraphs (P)

Paragraphs are used to define a block of text that forms a coherent thought or idea.

Characteristics :-

1. Displayed in a standard font size and weight
2. Typically used for body text, articles and content
3. can contain text, images, links, and other inline elements.
4. Automatically adds a line break after the paragraph.

Example :-

<P> This is a paragraph of text. It can contain multiple sentences. </P>

key differences :-

1. Purpose :- Headings define titles, while paragraphs define body text.

~~Keyedit~~

2. Font size and weight :- Heading are larger and bolder while paragraphs are standard.

3. Structure :- Headings are used to create a hierarchical structure, while paragraphs are used to present continuous text.

4. Search engine optimization (SEO) : Heading are more important for SEO than paragraphs.

★ Best Practices

1. use ~~only~~ headings to create a clear hierarchical structure.

2. use paragraphs to separate blocks of text.

3. Avoid using headings for styling text; use CSS instead

4. use only one h1 per page, and use subsequent headings (h2 - h6) to create subheadings.

By understanding the differences between headings and paragraphs, you can create well-structured readable and accessible HTML documents.

Q.2 How can you create a horizontal rule (<hr> tag) in HTML? Describe its Purpose and Provide an example of its usage.

Answer Purpose :-

The <hr> tag serves several Purposes :-

1. visual separation :- Divides content into distinct sections.

2. structural separation :- Indicates a break in thought or them.

3. layout control : Helps organize content and crete - whitespace.

syntax :

<hr>

Attributes :-

1. align : (deprecated) : left, righ, center
2. size : (deprecated) : specifies line thickness (e.g. size="5")
3. width : (deprecated) : specifies line width (e.g. width="50"%)

Note :- The align, size, width, and color attributes are deprecated in HTML 5

★ CSS Styling :

```
hr {
  border: none;
  border-top: 1px solid #ccc;
  width: 50%;
  margin: 20px auto;
}
```

Example usage :-

```
<h1> Introduction </h1>
<p> welcome to our website. </p>
<hr>
<h2> features </h2>
<p> our product offers... </p>
<hr>
<h3> contact us </h3>
<p> get in touch with us. </p>
```

in this example, the <hr> tag separates the introduction, features and contact sections.

★ Best practies :

1. use <hr> sparingly to avoid clutter.
2. use CSS to customize appearance.
3. ensure accessibility by providing alternative text for screen readers.

Semantic Alternatives :-

Consider using semantic elements instead of <hr>;

1. <section> : Defines a self-contained section.

2. <article> : Defines an independent piece of content

3. : with CSS : creates a styled container

By using the <hr> tag effectively, you can enhance your website's readability and structure.

Q.3 Discuss the use of subscript <sub> and superscript <sup> tags in HTML. Provide examples demonstrating their applications in text formating.

Answer Subscript (<sub>)

The <sub> tag reduces the font size and lower the baseline of the text, making it appear as subscript.

Example :

H2o (water) <sub> 2K/sub>

Result : H_2O (water)

Applications :

1. chemical formulas (e.g. H2O)
2. mathematical expressions (e.g. x3)
3. Footnotes or citations

superscript (<sup>)

The <sup> tag increases the font size and raises the baseline of the text, making it appear as superscript

example :-

2³=8

Result: $2^3 = 8$

Applications :-

1. mathematical expressions (e.g. 2^3)
2. Exponents (e.g. x^2)
3. Dates (e.g. 24 th January)
4. copyright symbols (e.g. © copy: ²⁰²²)

comparison :-

Tag	Effect	Example
<sub>	subscript	H2O (water) ₂
<sup>	superscript	2³ = 8

Styling Alternatives :-

instead of using <sub> and <sup> tags, you can use CSS to achieve similar effects :

- subscript :- Font-size : smaller : vertical-align : sub ;
- superscript :- font-size : larger : vertical-align : super ;

Best practices :-

1. use <sub> and <sup> tags for semantic meaning, not just visual styling.

2. use CSS for custom styling and layout control.

Browser support :-

Both <sub> and <sup> tags are widely supported across modern browsers, including.

- Google chrome
- mozilla firefox
- safari
- microsoft edge.

By using the <sub> and <sup> tags effectively, you can enhance your website's text formatting.

Q.5 compare and contrast unordered lists (<ul>), ordered lists (<ol>), and definition list (<dl>) in HTML

Answer unordered lists (<ul>)

- Purpose :- Present a list of items without numerical ordering.

- characteristics:
- Bullet Points (dics, circle, or square)
- No inherent order
- items are enclosed in <li> tags.

- Example

```
<ul>
  <li> Apple <li/>
  <li> Banana <li/>
  <li> cherry <li/>
<l/ul>
```

- Result:

- Apple
- Banana
- cherry

ordered list (<ol>)
- Purpose : Present a list of items with numberical ordering.

- characteristics

- Numbered list items (1, 2, 3 etc)
- items are enclosed in <li> tags.
- supports start attribute to specify starting number.

* Example

```
<ol>
  <li> Apple </li>
  <li> Banana </li>
  <li> cherry </li>
</ol>
```

Result:

1. Apple
2. Banana
3. cherry

Definition lists (<dl>)

- Purpose : Present a list of terms with definitions.
- characteristics :-
  - Consists of <dt> (term) and <dd> (definition) tags.
  - No inherent order
  - supports multiple definitions per term.

- Example :-

```
<dl>
  <dt> Fruit </dt>
  <dd> A sweet, edible Plant Product. </dd>
  <dt> vegetable </dt>
  <dd> A Plant Product used in cooking </dd>
</dl>
```

Result

Fruit
: A sweet, edible Plant Product
vegetable
: A Plant Product used in cooking

Comparison

list type	Purpose	ordering	Tags
<ul>	unordered	No order	<li>
<ol>	ordered	Numerical	<li>
<dl>	Definition	No order	<dt>, <dd>

contrast

- <ul> and <ol> use <li> tags, while <dl> uses <dt> and <dd> tags.

- <ol> has inherent numerical ordering while <ul> and <dl>

do not

- <dl> supports multiple definitions per term, while <ul> and <ol> do not

Best Practices:

1. use <ul> for lists without numerical ordering.

2. use <ol> for lists with numerical ordering.

3. use <dl> for definition lists.

4. Ensure accessibility by providing alternative text for screen readers.

- By understanding the differences between unordered list, ordered lists, and definition lists, you can effectivly present information on your website.

Assignment No :- 03

[unit :- 03]

Q.1 Explain the structure of an HTML table. Provide an example of a basic structure.

Answer

Basic structure :-

```
<table>
  <tr>
    <th> Header 1 </th>
    <th> Header 2 </th>
  </tr>
  <tr>
    <td> Data 1 </td>
    <td> Data 2 </td>
  </tr>
  <tr>
    <td> Data 3 </td>
    <td> Data 4 </td>
  </tr>
</table>
```

Elements :-

1. <table> : Defines the table.
2. <tr> : Defines a table row.
3. <th> : Defines a table header cell.

* Table Structure :-

1. Header Row :- The first row, containing <th> elements defines the column headers

2. Data Rows :- subsequent rows, containing <td> elements define the data.

3. columns :- Each <th> or <td> element represents a columns.

4. ~~columns~~ cells : Each <th> or <td> element represents a single cell.

Attributes :-

1. border : specifies the border with and style.

2. cellpadding : specifies the space between cell content and borders

3. width and height :- specify the table dimensions.

Example with Attributes :

<table border = "1" cellpadding = "5"
cellspacing "a" width "50%">

```
<tr>
  <th> Header 1 <\th>
  <th> Header 2<\th>
<\tr>
<tr>
  <td> Data 1 <\td>
  <td> Data 2<\td>
<\tr>
<tr>
  <td> Data 3 <\td>
  <td> Data 4 <\td>
<\tr>
<\table>
```

Additional Elements :-

1. <caption> : Defines the table caption.

2. <col> and <colgroup> : Define groups and styles

Best Practices :-

1. use semantic HTML table structure.
2. use CSS for styling and layout control.
3. Ensure accessibility by providing alternative text for screen readers.
4. use scope attribute to define header scope.

By understanding the basic structure and elements of an HTML table, you can crete well-structured and accessible table.

Q.2 How can you align a table and its cell contents in HTML? Describe the CSS properties or HTML attributes used for alignment and provide example demonstrating their application.

Answer:- HTML Atributes (Deprecated):

Although deprecated, some HTML attributes are still supported for backward compatibility.

- align (left, right, center)
- valign (top, middle, bottom)
- char (character, alignment)
- charoff (character offset)

Example :-

```
<table align="center">
<tr>
<td valign="top">Top-aligned</td>
<td
valign="middle">middle-aligned</td>
<td
valign="bottom">Bottom-aligned</td>
</tr>
</table>
```

CSS Properties :

Recommended approach using CSS Properties

* text-align (left, right, center, justify)
* vertical-align (top, middle, bottom, baseline)
* padding (cell padding)

Examples :-

Table Alignment :

```
Table {
    margin : 0 auto ; /* center table
horizontally */
}
```

Cell content Alignment :

```
td {
    text-align : center; /* Horizontal alignment */
    vertical-align : middle; /*
vertical alignment */
}
```

Specific cell Alignment :

```
td : nth-child (1) {
    ~~tex~~ text-align : left;
    vertical-align : top;
}
```

using CSS classes:

```
• centered-table {
margin: 0 auto;
}
• centered-cell {
text-align: center;
vertical-align: middle;
}

<table class="centered-table">
<tr>
<td
class="centered-cell">centered cell</td>
</tr>
</table>
```

Best Practices:

1. use CSS Properties instead of HTML attributes.
2. Define table and cell style in External CSS Files.
3. use semantic html table structure,
4. Ensure accessibility by Providing alternative text for screen reader.

By applying these alignment techniques you can effectively control the layout of your tables and cell contents.

Q.3 Describe the process of nesting tables in HTML. Provide an example illustrating nested tables.

Answer: Process:

1. Create the outer table with <table>, <tr>, and <td> elements.

2. within a table cell (<td>) add another <table> element.

3. Define the inner table's structure with <tr> and <td> elements.

4. Close the inner table (</table>) within the outer table cell.

5. close the outer table (</table>).

Example:

```
<table border="1">
<tr>
    <th> Name </th>
    <th> Address </th>
</tr>
<tr>
    <td> John Doe </td>
<td>
    <table border="1">
    <tr>
```

```
<th> street </th>
<th> city </th>
<th> state </th>
</tr>
<tr>
<td> 123 main st </td>
<td> Anytown </td>
<td> CA </td>
</tr>
</table>
</td>
</tr>
<tr>
<td> June smith </td>
<td>
<table border = "1">
<tr>
<th> street </th>
<th> city </th>
<th> state </th>
</tr>
<tr>
<td> 456 Elm st </td>
<td> othertown </td>
<td> NY </td>
</tr>
</table>
</td>
</tr>
</table>
```

Result: A table with columns, where the second column contains another table with three columns.

Tips :-

1. use semantic HTML table structure.
2. Ensure accessibility by providing alternative text for screen readers.
3. use CSS for styling and layout control.
4. Avoid excessive nesting as it can affect readability

common use cases:

1. Displaying complex data relationships
2. creating master - details views.
3. Organizing hierarchical information.

Best Practices:

1. Keep nested table simple and readable.
2. use clear and concise table headers.
3. Avoid using nested tables for layout purposes.

By nesting tables effectively, you can present complex data in a clear and organized manner.

Q.4 what are frames in HTML, and how are they created? Explain the purpose of the <Frameset> and <Frame> tags.

Answer Process :-

1. Create the outher table with <table>, <tr>, and <td> elements.

2. within a table cell (<td>), add another <table> element.

3. Define the inner table's structure with <tr> and <td> elements.

4. Close the outher table (</table>)

Example :-

```
<table border="1">
<tr>
  <th> Name </th>
  <th> Address </th>
</tr>
<tr>
  <td> John Doe </td>
  <td>
  <table border="1">
  <tr>
  <th> street </th>
```

```
<th> city <|th>
<th> state <|th>
<|tr>
<tr>
  <td> 123 main st <|td>
  <td> CA <|td>
<|tr>
<|table>
<|td>
<|tr>
<tr>
  <td> Jane Smith <|td>
  <td>
  <table border = "1">
  <tr>
    <th>
    ~~<table~~ street <|th>
    <th> city <|th>
    <th> state <|th>
<|tr>
<tr>
  <td> 456 elm st <|td>
  <td> other town <|td>
<|tr>
<|table>
<|td>
<|tr>
<|table>
```

Result:

A table with two column, where the second column contains another table with three columns.

Tips:

1. use semantic HTML table structure.
2. ensure accesibility by Providing alternative text for screen readers.
3. use CSS styling and layout control.
4. Avoid excessing nesting, as it can affect readity.

Common use cases:

1. Displaying complex data relationships.
2. creating master - details views.
3. organizing hierarchical information.

Best Practices:-

1. Keep nested tables simple and readable.
2. use clear and concise table headers.
3. Avoid using nested tables for layout Purposes.

By nesting tables effectively you can Present complex data in a clear and organized manner.

Additional Examples:

- Nested tables with multiple levels:

```
<table>
  <tr>
    <td>
      <table>
        <tr>
          <td>
            <table>
              <tr>
                <td> Deeply nested
table <td>
                </tr>
              </table>
            </td>
          </tr>
        </table>
      </td>
    </tr>
  </table>
```

- Nested tables with different orientations:

```
<table>
  <tr>
    <td>
      <table style="width: 100%">
```

```
<tr>
    <td> Horizontal nested
table <td>
    <tr>
    <table>
    <td>
  <tr>
  <tr>
    <td>
      <table style="heigh : 100%">
      <tr>
      <td> vertical nested
table <td>
    <tr>
    <table>
    <td>
  <tr>
  <table>
```

Q.5 Discuss the concept of applying hyperlink targets to frames.

Answer: Frame targets:

in HTML, you can specify a target attribute for hyperlink (<a>) to determine where the linked page opens.

1. blank: opens in a new window

2. self: opens in the same frame (default)

3. Parent : oppens in the parent frame.

4. top :- opens in the full browser window.

Example:

```
<a href="page.html"
target="mainframe">open in
mainframe</a>
```

Frames Names:

To use frame targets, you must assign names to your frames using the name attribute.

```
<frame src="page1.html"
  name="mainframe">
```

Targeting frames:

1. Intra-frame targeting : linking within the same frame set.

```
<a href="page2.html"
target="mainframe">open in
mainframe</a>
```

1 Intra - frame targeting : linking within the same frame set.

<a href="Page2.html"
target="sideBar"> open in sideBar frame </a>.

2. Parent frame targeting : linking to the parent frame.

<a href="Page2.html"
target="parent"> open in parent frame </a>

Best practices :-

1. use descriptive frame names
2. Ensure frame name match target attribute.
3. Test frame targeting thoroughly.

Alternatives to frame targets.

1. CSS layout
2. Javascript libraries (e.g AJax)
3. HTML5 semantic elements.

Browser support :

- google chrome
- safari

Assignment :- 4
unit :- 04

Q.1 Explain the different ways to create hyperlinks in HTML

Answer 1. Absolute URL Hyperlink

<a href="(link unavilable)"> visit Example </a>

2. Relative URL Hyperlink

<a href="about.html"> About us </a>

3. Anchor link (intra page linking)

<a href="#top"> Back to top </a>

4. Email link

<a href="mailto: info@Example.com") Email us </a>

5. Phone Number link

<a href="tel:+1234567890"> call us </a>

6. link to a specific section (ID)

<a href="#section"> section 1 </a>

```
<button
onclick="location.href='#'">click
Here</button>
```

image link

```
<a href="#"
  <img src="image.jpg" alt="Image Description">
</a>
```

★ Best Practices

1. use ~~use~~ descriptive link text.
2. ensure accessibility with alt and title attributes.
3. use relative URLs for internal links.
4. Test links thoroughly

★ Browser support

- google chrome
- mozilla Firefox
- safari
- microsoft Edge
- internet Explorer (IE+3)

Q.2 How can you style hyperlinks in HTML to change their color?

Answer inline styling

```
<a bref ="#" style ="colour:
blue;" > Blue link </a>
```

internal CSS

```
<head>
  <style>
    a {
      colour : red ;
    }
  </style>
  </head>
  <body>
      <a bref ="#" > Red link </a>
  </body>
```

External CSS

Create a CSS file (style.css) and link it to your HTML file:

Style.css

```
a {
  colour : green ;
}
```

index.html

```
<head>
```

```
<link rel="stylesheet"
type="text/css" href="style.css">
<head>
<body>
    <a href="#"> Green link </a>
</body>
```

Pseudo-classes

style different states of a hyperlink:

```
a:link {
    colour: blue; /* unvisited link */ }

a: visited {
    color: ~~red~~ Purple; /* visited link */ }

a: hover {
    colour: red; /* Hovered link */ }

a: active {
    color: green; /* Active link */ }
```

Class-Based Styling

Apply styles to specific hyperlinks:

```
<a href="#" class="blue-link"> Blue
link </a>
```

```
<style>
    .blue-link {
      color : blue;
    }
</style>
```

- Best Practices

1. use external css files for better organization.
2. use meaningful class names.
3. Avoid inline Styling
4. Test for accessibility.

Browser support

- google chrome
- mozilla firefox
- safari
- internet explorer etc.)

Q.3 Describe the Process of inserting an image on a webPage using the <img> tag in HTML.

Answer Basic syntax

```
<img src="image_uri"
alt="image_description">
```

Attributes:

1. src: specifies the image URL
2. alt: provides alternative text for accessibility.
3. width: height: Define image dimensions
4. title: Adds a tooltip with image information.

Example:

```
<img src="logo.png" alt="company
logo" width="200" height="100">
```

image sources:

1. local files: src="image.jpg"
2. web URLs: src="(link unavailable)"
3. Relative paths: src="../images/image.jpg"

images formats:

1. JPEG (.JPG, .JPeg)
2. PNG (.Png)
3. GIF (.gif)
4. SVG (.svg)
5. BMP (.bmp)

Best Practices:

1. use descriptive alt text.
2. optimize image file size

3. Specify images dimensions.
4. Use relative paths for local images.

★ Common issues :

1. Broken image links
2. image not displaying due to incorrect File Path
3. image dimensions not specified

★ Browser suport :

The <img> tag is supported by all modern browsers including :

- Google chrome
- safari
- internet explorer (IE 10+)

Additional TIPS :

1. use CSS to style images
2. use Javascript to create Interactive images.

Semantic HTML.

use the <figure> and <figcaption> tags to provide context for images.

conclusion.

Applying hypolink targets to frames enables precise control over where linked pages open within a framed website. understanding frame targets and their usage is essential for creating user-friendly and accessible framed websites.

SILVER OAK UNIVERSITY
EDUCATION TO INNOVATION

1. Flash Player
2. Third Party

Best Practices :-

1. use semantic Html Structure.

Additional Features.

1. Custom Controls using Javascript.
2. Adaptive bitrate streaming

Example with Subtitles :-

```
<video width = "640" height = "480" controls>
    <source src = "video.mp4"
  <track src = "subtitles.vtt"
kind = "subtitles" srclang = "en"
label = "English">

  Your browser does not support the video tag.
</video>
```

By using HTML5's multimedia tags, you can easily embed enguging content on your web pages.

Audio Tag :

<audio controls>
<source src="audio.mp3"
type="audio/mp3">
Your browser does not support the audio tag.
</audio>

Attributes :

1. controls : Display audio controls (play, pause)
2. loop : loop audio playback

Source options :

1. mp4 (video)
2. webm (video)
3. ogg (video and audio)
4. wav (audio)

Browser support

The <video> and <audio> tags are supported by

- Google Chrome
- Safari

Fallback options :

```
<figure>
    <img src="image.TPY" alt="Image
Description">
    <figcaption> Image caption </
figcaption>
</figure>
```

Q.5 explain how multimedia content can be embedded on a webpage using HTML5 <video> and <audio> tags.

Answers Video Tag :

```
<video width="640" height="480"
Controls>
    <source src="video.mp4"
type="vido/mp4">
    Your browser does not support the video tag.
</video>
```

~~Ans~~ Attributes :-

1. width and heigh : Define vid o dimensions

2. controls : Display video controls (play, pause, etc)

3. autoplay : Play video automatically

4. loop : loop video playback