***List of HTML tags in HTML5***

1. `<html>` {Container Tag} </html>

- Purpose: Defines the root of an HTML document.

- Example: `<html> ... </html>`

- Rules: The entire HTML document structure is contained within `<html>` and `</html>` tags.

2. `<head>` {Container Tag} </head>

- Purpose: Contains meta-information about the document.

- Example: `<head> ... </head>`

- Rules: Should include elements like `<title>`, `<meta>`, `<link>`, `<style>`, and `<script>`.

3. `<title>` </title>

- Purpose: Sets the title of the document, displayed in the browser's title bar.

- Example: `<title>Page Title</title>`

- Rules: Must be placed within the `<head>` section.

4. `<body>` {Container Tag} </body>

- Purpose: Contains the content of the document visible to users.

- Example: `<body> ... </body>`

- Rules: All visible content like text, images, links, etc., should be placed within `<body>` tags.

5. `<h1>` to `<h6>` </h1> to </h6>

- Purpose: Defines headings of different levels.

- Example: `<h1>Heading 1</h1>`

- Rules: Used to structure content with hierarchy, where `<h1>` is the highest level and `<h6>` is the lowest.

6. `<p>` </p>

- Purpose: Defines a paragraph.

- Example: `<p>This is a paragraph.</p>`

- Rules: Used to structure textual content into paragraphs.

7. `<a>` </a>

- Purpose: Defines a hyperlink.

- Example: `<a href="https://example.com">Link Text</a>`

- Rules: Requires an `href` attribute specifying the URL of the link.

8. `<img>` (Self-closing)

- Purpose: Embeds an image in the document.

- Example: `<img src="image.jpg" alt="Image Description">`

- Rules: Must have a `src` attribute specifying the image URL. The `alt` attribute provides alternative text for accessibility.

9. `<div>` {Container Tag} </div>

- Purpose: Defines a division or section in an HTML document.

- Example: `<div>This is a division.</div>`

- Rules: Used for grouping and applying styles to sections of content.

10. `<span>` </span>

- Purpose: Defines a span of text.

- Example: `<span style="color: red;">Red Text</span>`

- Rules: Typically used for styling or targeting specific sections of text.

11. `<ul>` {Container Tag} </ul>

- Purpose: Defines an unordered list.

- Example: `<ul> <li>Item 1</li> <li>Item 2</li> </ul>`

- Rules: Contains one or more `<li>` (list item) elements.

12. `<ol>` {Container Tag} </ol>

- Purpose: Defines an ordered list.

- Example: `<ol> <li>Item 1</li> <li>Item 2</li> </ol>`

- Rules: Contains one or more `<li>` (list item) elements.

13. `<li>` </li>

- Purpose: Defines a list item.

- Example: `<ul> <li>Item 1</li> <li>Item 2</li> </ul>`

- Rules: Must be a child of `<ul>` or `<ol>` tags.

14. `<table>` {Container Tag} </table>

- Purpose: Defines a table.

- Example: `<table> ... </table>`

- Rules: Contains `<tr>` (table row) elements.

15. `<tr>` {Container Tag} </tr>

- Purpose: Defines a table row.

- Example: `<tr> ... </tr>`

- Rules: Contains one or more `<td>` or `<th>` elements.

16. `<td>` </td>

- Purpose: Defines a table cell.

- Example: `<td>Cell Content</td>`

- Rules: Must be a child of a `<tr>` element.

17. `<th>` </th>

- Purpose: Defines a table header cell.

- Example: `<th>Header Cell</th>`

- Rules: Typically used within the first row (`<tr>`) of a table to denote header cells.

18. `<form>` {Container Tag} </form>

- Purpose: Defines an HTML form for user input.

- Example: `<form action="/submit-form" method="post"> ... </form>`

- Rules: Contains input elements like `<input>`, `<button>`, `<select>`, etc., for user interaction.

19. `<input>` (Self-closing)

- Purpose: Defines an input control.

- Example: `<input type="text" name="username">`

- Rules: Various types available (text, password, checkbox, radio, etc.) specified via the `type` attribute.

20. `<button>` {Container Tag} </button>

- Purpose: Defines a clickable button.

- Example: `<button>Click Me</button>`

- Rules: Can contain text or other HTML elements.

21. `<select>` {Container Tag} </select>

- Purpose: Defines a dropdown list.

- Example: `<select> <option value="1">Option 1</option> <option value="2">Option 2</option> </select>`

- Rules: Contains one or more `<option>` elements.

22. `<option>` </option>

- Purpose: Defines an option in a dropdown list.

- Example: `<option value="1">Option 1</option>`

- Rules: Must be a child of a `<select>` element.

23. `<textarea>` {Container Tag} </textarea>

- Purpose: Defines a multiline input control.

- Example: `<textarea rows="4" cols="50"></textarea>`

- Rules: Specifies the number of rows and columns for the textarea.

24. `<iframe>` (Self-closing)

- Purpose: Embeds another HTML page into the current page.

- Example: `<iframe src="page.html"></iframe>`

- Rules: Requires a `src` attribute specifying the URL of the page to embed.

25. `<audio>` {Container Tag} </audio>

- Purpose: Embeds audio content into the document.

- Example: `<audio src="audio.mp3" controls></audio>`

- Rules: Requires a `src` attribute specifying the URL of the audio file.

26. `<video>` {Container Tag} </video>

- Purpose: Embeds video content into the document.

- Example: `<video src="video.mp4" controls></video>`

- Rules: Requires a `src` attribute specifying the URL of the video file.

27. `<canvas>` {Container Tag} </canvas>

- Purpose: Defines an area for drawing graphics using JavaScript.

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

- Rules: Requires `width` and `height` attributes to specify the dimensions of the canvas.

28. `<svg>` {Container Tag} </svg>

- Purpose: Defines scalable vector graphics.

- Example: `<svg width="100" height="100"> ... </svg>`

- Rules: Contains various elements for drawing shapes, paths, text, etc., using XML syntax.

29. `<script>` </script>

- Purpose: Embeds JavaScript code into the document.

- Example: `<script>console.log("Hello, World!");</script>`

- Rules: Can be placed in the `<head>` or `<body>` section of the document.

30. `<style>` </style>

- Purpose: Defines CSS styles for the document.

- Example: `<style> body { background-color: lightblue; } </style>`

- Rules: Can be placed in the `<head>` section of the document.

Of course, let's format the information as requested:

31. `<nav>` {Container Tag} </nav>

- Purpose: Defines navigation links.

- Example:

```html

<nav>

<a href="/">Home</a>

<a href="/about">About</a>

<a href="/contact">Contact</a>

</nav>

```

- Rules: Typically used to wrap sets of navigation links.

32. `<header>` {Container Tag} </header>

- Purpose: Defines introductory content or a group of introductory elements.

- Example:

```html

<header>

<h1>Website Title</h1>

<p>Welcome to our website!</p>

</header>

```

- Rules: Often contains headings, logos, introductory text, etc.

33. `<footer>` {Container Tag} </footer>

- Purpose: Defines a footer for a document or section.

- Example:

```html

<footer>

<p>&copy; 2024 Example Company</p>

</footer>

```

- Rules: Typically used to contain copyright information, contact details, etc.

34. `<article>` {Container Tag} </article>

- Purpose: Defines an independent piece of content that can stand alone.

- Example:

```html

<article>

<h2>Article Title</h2>

<p>Article content goes here...</p>

</article>

```

- Rules: Represents self-contained content such as blog posts, news articles, etc.

35. `<section>` {Container Tag} </section>

- Purpose: Defines a section in a document.

- Example:

```html

<section>

<h2>Section Title</h2>

<p>Section content goes here...</p>

</section>

```

- Rules: Used to group related content together.

36. `<aside>` {Container Tag} </aside>

- Purpose: Defines content aside from the content it is placed in.

- Example:

```html

<aside>

<h3>Related Links</h3>

<ul>

<li><a href="/related1">Related Page 1</a></li>

<li><a href="/related2">Related Page 2</a></li>

</ul>

</aside>

```

- Rules: Often used for sidebars or secondary content.

37. `<main>` {Container Tag} </main>

- Purpose: Defines the main content of a document.

- Example:

```html

<main>

<h1>Main Content</h1>

<p>Main content goes here...</p>

</main>

```

- Rules: Should contain the primary content of the document, excluding headers, footers, and navigation.

38. `<figure>` {Container Tag} </figure>

- Purpose: Specifies self-contained content, like images, diagrams, illustrations, etc., with a caption.

- Example:

```html

<figure>

<img src="image.jpg" alt="Image Description">

<figcaption>Image Caption</figcaption>

</figure>

```

- Rules: Used to group media content with its caption.

39. `<figcaption>` </figcaption>

- Purpose: Defines a caption for a `<figure>` element.

- Example:

```html

<figure>

<img src="image.jpg" alt="Image Description">

<figcaption>Image Caption</figcaption>

</figure>

```

- Rules: Must be placed immediately after the content it is captioning within a `<figure>` element.

40. `<details>` {Container Tag} </details>

- Purpose: Defines additional details that the user can view or hide.

- Example:

```html

<details>

<summary>Click to view details</summary>

<p>Additional details go here...</p>

</details>

```

- Rules: Contains a `<summary>` element followed by the details to be shown or hidden.

41. `<summary>` </summary>

- Purpose: Defines a visible heading for a `<details>` element.

- Example:

```html

<details>

<summary>Click to view details</summary>

<p>Additional details go here...</p>

</details>

```

- Rules: Should be the first child of a `<details>` element and serves as the visible heading.

42. `<mark>` </mark>

- Purpose: Highlights text within a document.

- Example:

```html

<p>This is <mark>highlighted</mark> text.</p>

```

- Rules: Used to visually distinguish text from surrounding content.

43. `<time>` </time>

- Purpose: Represents a specific period in time.

- Example:

```html

<p>Today is <time datetime="2024-04-25">April 25, 2024</time>.</p>

```

- Rules: The `datetime` attribute should contain a valid date and time in the format `YYYY-MM-DD`.

44. `<progress>` </progress>

- Purpose: Represents the progress of a task.

- Example:

```html

<progress value="70" max="100"></progress>

```

- Rules: The `value` attribute indicates the progress made, while the `max` attribute specifies the maximum value.

45. `<meter>` </meter>

- Purpose: Represents a scalar measurement within a known range.

- Example:

```html

<meter value="3" min="0" max="10">3 out of 10</meter>

```

- Rules: The `value`, `min`, and `max` attributes define the current value, minimum value, and maximum value respectively.

46. `<ruby>` {Container Tag} </ruby>

- Purpose: Defines a ruby annotation, which is used to provide pronunciation or meaning for characters.

- Example:

```html

<ruby>

漢 <rp>(</rp><rt>Kan</rt><rp>)</rp>

字 <rp>(</rp><rt>ji</rt><rp>)</rp>

</ruby>

```

- Rules: Contains one or more `<rb>` (ruby base) elements and `<rt>` (ruby text) elements.

47. `<rt>` </rt>

- Purpose: Defines the pronunciation or meaning of characters in a ruby annotation (used in conjunction with `<ruby>`).

- Example:

```html

<ruby>

漢 <rp>(</rp><rt>Kan</rt><rp>)</rp>

字 <rp>(</rp><rt>ji</rt><rp>)</rp>

</ruby>

```

48. `<rp>` </rp>

- Purpose: Provides fallback parentheses for browsers that do not support ruby annotations.

- Example:

```html

<ruby>

漢 <rp>(</rp><rt>Kan</rt><rp>)</rp>

字 <rp>(</rp><rt>ji</rt><rp>)</rp>

</ruby>

```

- Rules: Should contain opening and closing parentheses for ruby annotations.

49. `<dialog>` {Container Tag} </dialog>

- Purpose: Defines a dialog box or window for user interaction.

- Example:

```html

<dialog open>

<p>Dialog content goes here...</p>

<button>Close</button>

</dialog>

```

- Rules: The `open` attribute indicates that the dialog box is open by default.

50. `<template>` {Container Tag} </template>

- Purpose: Defines HTML content that should be hidden when the page loads, but can be instantiated later on.

- Example:

```html

<template>

<p>This content will be hidden initially.</p>

</template>

```

- Rules: The content inside `<template>` tags is inert and not rendered until cloned via JavaScript.

These are the additional minor HTML5 tags along with their purposes, examples, and usage rules. Let me know if you need further assistance!