**Form all html tag**

A <form> element is the main container for a web form, which is used to collect user inputs. HTML provides various tags within <form> to facilitate different types of input fields, grouping, and interactive elements. Here’s a list of commonly used HTML tags within a <form>:

### Core Form Tags

1. **<form>**:
   * The container for all form elements.
   * Attributes: action, method, enctype, name, target, autocomplete, etc.
2. **<input>**:
   * Creates different types of input fields like text, password, checkbox, radio, etc.
   * Common attributes: type, name, value, placeholder, required, readonly, disabled, maxlength, etc.
3. **<label>**:
   * Provides a label for form inputs, making forms more accessible.
   * Attributes: for (associates the label with an id of an input element).
4. **<textarea>**:
   * Allows users to enter multi-line text.
   * Attributes: name, rows, cols, maxlength, placeholder, etc.
5. **<button>**:
   * Represents a clickable button, often used to submit or reset the form.
   * Attributes: type (submit, reset, button), name, value.
6. **<select>**:
   * Creates a dropdown menu.
   * Attributes: name, size, multiple.
7. **<option>**:
   * Defines individual items within a <select> dropdown.
   * Attributes: value, selected.
8. **<optgroup>**:
   * Groups related <option> items within a <select> element.
   * Attributes: label, disabled.
9. **<fieldset>**:
   * Groups related elements within a form, visually separating them.
   * Attributes: disabled, name.
10. **<legend>**:
    * Provides a caption for a <fieldset>, describing its purpose.
11. **<datalist>**:
    * Specifies a list of pre-defined options for an <input> element (typically type="text").
    * Works in conjunction with the list attribute on <input>.
12. **<keygen>** (Deprecated):
    * Used to generate a public-private key pair for form-based encryption.
13. **<output>**:
    * Represents the result of a calculation or user action, often dynamically updated.
    * Attributes: for, name.
14. **<progress>**:
    * Displays the progress of a task (like a loading bar).
    * Attributes: value, max.
15. **<meter>**:
    * Displays a scalar measurement within a known range (like a thermometer or rating scale).
    * Attributes: value, min, max, low, high, optimum.

**Attributes of the <form> Element**

1. **action**:
   * Specifies the URL where the form data should be sent for processing.
   * Example: <form action="/submit\_form">
2. **method**:
   * Defines the HTTP method to be used when sending form data. Common values are:
     + GET: Sends data in the URL (query string). Suitable for data retrieval.
     + POST: Sends data in the request body. Used for secure or larger data submissions.
   * Example: <form method="POST">
3. **enctype**:
   * Specifies the encoding type of the form data when using POST. Common values are:
     + application/x-www-form-urlencoded: Default encoding for form data.
     + multipart/form-data: Used when the form contains file uploads.
     + text/plain: Sends data without encoding, generally not recommended.
   * Example: <form enctype="multipart/form-data">
4. **target**:
   * Specifies where to display the response after the form is submitted. Common values include:
     + \_self: Opens in the same window (default).
     + \_blank: Opens in a new tab or window.
     + \_parent: Opens in the parent frame.
     + \_top: Opens in the full body of the window.
   * Example: <form target="\_blank">
5. **autocomplete**:
   * Controls whether the browser should enable autocomplete for form fields. Options are:
     + on: Allows autocomplete (default).
     + off: Disables autocomplete.
   * Example: <form autocomplete="off">
6. **novalidate**:
   * Disables HTML5 form validation. Useful if you prefer to validate the form data via JavaScript.
   * Example: <form novalidate>
7. **name**:
   * Specifies a name for the form, which can be used to identify the form in JavaScript.
   * Example: <form name="registrationForm">
8. **rel**:
   * Used with the target attribute to define the relationship between the current document and the response. Commonly used values:
     + noopener: Prevents the new page from accessing the original window.
     + noreferrer: Prevents referrer information from being sent with the request.
   * Example: <form target="\_blank" rel="noopener">

**Common Attributes of <input>**

1. **type**:
   * Specifies the type of input, such as text, password, email, number, checkbox, radio, file, submit, etc.
   * Example: <input type="text">
2. **name**:
   * Defines the name of the input, which is sent to the server as the key when the form is submitted.
   * Example: <input type="text" name="username">
3. **value**:
   * Sets the initial value of the input field. For submit buttons, it defines the button text.
   * Example: <input type="text" value="John Doe">
4. **placeholder**:
   * Displays placeholder text inside the input when it’s empty, providing a hint to the user.
   * Example: <input type="text" placeholder="Enter your name">
5. **required**:
   * Specifies that the input must be filled out before submitting the form.
   * Example: <input type="text" required>
6. **readonly**:
   * Makes the input field non-editable. The user can see the value but cannot change it.
   * Example: <input type="text" value="Fixed Text" readonly>
7. **disabled**:
   * Disables the input, making it unclickable and not included in form submission.
   * Example: <input type="text" disabled>
8. **maxlength**:
   * Limits the maximum number of characters allowed in the input.
   * Example: <input type="text" maxlength="10">
9. **min and max**:
   * Set minimum and maximum values for numeric inputs.
   * Example: <input type="number" min="1" max="100">
10. **step**:
    * Specifies the increment for numeric inputs.
    * Example: <input type="number" step="5">
11. **pattern**:
    * Defines a regular expression pattern that the input value must match for validation.
    * Example: <input type="text" pattern="[A-Za-z]{3}">
12. **autocomplete**:
    * Controls whether the browser should autocomplete the input field based on previous entries.
    * Example: <input type="text" autocomplete="off">
13. **autofocus**:
    * Automatically focuses the input field when the page loads.
    * Example: <input type="text" autofocus>
14. **minlength**:
    * Specifies the minimum number of characters allowed in the input.
    * Example: <input type="text" minlength="5">
15. **size**:
    * Defines the visual width of the input field in characters.
    * Example: <input type="text" size="20">
16. **multiple**:
    * Allows multiple values, typically used with file and email types.
    * Example: <input type="file" multiple>
17. **list**:
    * Associates the input with a <datalist> element for predefined options.
    * Example:

html

Copy code

<input type="text" list="suggestions">

<datalist id="suggestions">

<option value="Option 1">

<option value="Option 2">

</datalist>

1. **form**:
   * Associates the input with a specific form id, allowing it to be outside the <form> element.
   * Example: <input type="text" form="myForm">

**Attributes of <label>**

The <label> element has only one specific attribute:

1. **for**:
   * The for attribute links the label to a form element by using the form element's id. This makes the label interactive, so clicking on the label will focus on or activate the associated form control.
   * Example: <label for="cars">Choose a car:</label>

### Common Attributes of <select>

1. **id**: Sets a unique identifier for the <select> element, making it accessible for styling and JavaScript.

html

Copy code

<select id="car"></select>

1. **name**: Defines the name of the <select> element. This is crucial for form data submission, as it becomes the key when submitting form data.

html

Copy code

<select name="cars"></select>

1. **size**: Specifies the number of visible options in the dropdown. By default, the dropdown will show one option, but setting size="4" will make it a listbox with four visible options.

html

Copy code

<select size="4"></select>

1. **multiple**: Allows multiple selections within the dropdown. Users can select multiple options by holding down the Ctrl or Shift key.

html

Copy code

<select multiple></select>

1. **disabled**: Disables the dropdown, preventing user interaction and submission.

html

Copy code

<select disabled></select>

1. **required**: Specifies that selecting an option is required for form submission. It prompts the user to select an option if left unselected.

html

Copy code

<select required></select>

1. **autofocus**: Automatically focuses the <select> element when the page loads.

html

Copy code

<select autofocus></select>

1. **form**: Associates the <select> element with a form. This is helpful if the <select> element is outside the <form> tag, linking it back to the form with a specific id.

html

Copy code

<select form="myForm"></select>

1. **class**: Specifies one or more class names for the <select> element, allowing for CSS styling.

html

Copy code

<select class="dropdown-style"></select>

**Attributes of the <option> Element**

1. **value**:
   * Specifies the value sent to the server when the option is selected. If omitted, the content between the <option> tags is used as the value.
   * Example: <option value="car">Car</option>
2. **selected**:
   * Indicates that the option should be preselected when the page loads. Only one option can be selected by default unless it is part of a multiple-selection list.
   * Example: <option value="bike" selected>Bike</option>
3. **disabled**:
   * Disables the option so it cannot be selected. Disabled options appear grayed out.
   * Example: <option value="plane" disabled>Plane</option>
4. **label**:
   * Provides an alternative text label for the option, which is useful for accessibility. If used, this label appears in place of the option text in the <select> dropdown.
   * Example: <option value="train" label="Train Option">Train</option>

**<optgroup> Attributes:**

1. **label**:
   * Defines the label for the group of <option> elements within the <optgroup>. It is a required attribute.
   * **Example**: <optgroup label="Fruits">
2. **disabled**:
   * If present, it disables the entire group of options within the <optgroup>. When disabled, none of the options within that group can be selected.
   * **Example**: <optgroup label="Fruits" disabled>

**Attributes of <textarea>:**

1. **rows**:
   * Specifies the number of visible text lines in the <textarea>.
   * **Example**: <textarea rows="4"></textarea>
2. **cols**:
   * Specifies the visible width of the <textarea> in terms of the number of characters.
   * **Example**: <textarea cols="50"></textarea>
3. **name**:
   * Defines the name of the <textarea>. This is important for form submission, as the value will be associated with this name.
   * **Example**: <textarea name="comment"></textarea>
4. **placeholder**:
   * Displays a short hint inside the <textarea> when it's empty, helping users understand what kind of input is expected.
   * **Example**: <textarea placeholder="Enter your comment here..."></textarea>
5. **maxlength**:
   * Specifies the maximum number of characters allowed in the <textarea>.
   * **Example**: <textarea maxlength="200"></textarea>
6. **readonly**:
   * Specifies that the <textarea> is read-only, meaning the user can see the text but cannot modify it.
   * **Example**: <textarea readonly>This is read-only text.</textarea>
7. **disabled**:
   * Disables the <textarea>, making it unresponsive to user input.
   * **Example**: <textarea disabled>This field is disabled.</textarea>
8. **wrap**:
   * Defines how the text should wrap in the <textarea>. The values are:
     + soft (default): The text will wrap visually, but the form data will contain the text as a single line.
     + hard: The text will be wrapped with line breaks when submitted.
   * **Example**: <textarea wrap="hard"></textarea>
9. **autofocus**:
   * Automatically focuses on the <textarea> when the page loads.
   * **Example**: <textarea autofocus></textarea>
10. **form**:
    * Associates the <textarea> with a specific <form> element (useful if the <textarea> is not inside a form element but you still want it to be part of a specific form submission).
    * **Example**: <textarea form="myform"></textarea>
11. **spellcheck**:
    * Enables or disables the spellcheck feature for the <textarea>. When set to "true", it highlights misspelled words.
    * **Example**: <textarea spellcheck="true"></textarea>
12. **wrap** (mentioned earlier, also applicable in mobile views):
    * Defines how the input text is wrapped (especially on smaller screens).

**<button> Attributes :**

1. **type**:
   * Specifies the type of button. There are three possible values:
     + submit: The button submits the form (default behavior for a button inside a form).
     + reset: The button resets the form to its default values.
     + button: The button does nothing by default. It's often used for custom JavaScript functionality.
   * **Example**: <button type="submit">Submit</button>
2. **name**:
   * Defines the name of the button. This can be useful when submitting forms, as the name-value pair is sent with the form data.
   * **Example**: <button name="submitButton">Submit</button>
3. **value**:
   * Defines the value associated with the button, often used with form submission to send data when the button is clicked.
   * **Example**: <button type="submit" value="submitForm">Submit</button>
4. **disabled**:
   * Disables the button, making it unclickable and preventing any user interaction with it.
   * **Example**: <button disabled>Submit</button>
5. **autofocus**:
   * Automatically focuses on the button when the page loads.
   * **Example**: <button autofocus>Click Me</button>
6. **form**:
   * Specifies the form that the button is associated with. This is useful when the button is placed outside of the <form> but still needs to submit or reset the form.
   * **Example**: <button form="myForm">Submit</button>
7. **formaction**:
   * Specifies the URL to send the form data to when the button is clicked. This attribute overrides the form’s action attribute.
   * **Example**: <button type="submit" formaction="submitForm.php">Submit</button>
8. **formenctype**:
   * Specifies the encoding type for the form data when the button is used to submit the form. This is commonly used with file uploads.
   * **Example**: <button type="submit" formenctype="multipart/form-data">Submit</button>
9. **formmethod**:
   * Specifies the HTTP method (GET or POST) to be used when the form is submitted via the button.
   * **Example**: <button type="submit" formmethod="POST">Submit</button>
10. **formnovalidate**:
    * Disables form validation when the form is submitted using this button. Typically used for buttons that submit data but don’t require validation.
    * **Example**: <button type="submit" formnovalidate>Submit</button>
11. **formtarget**:
    * Specifies where to display the response after submitting the form. It can have values like \_self, \_blank, \_parent, or \_top.
    * **Example**: <button type="submit" formtarget="\_blank">Submit</button>
12. **accesskey**:
    * Specifies a keyboard shortcut to activate or focus the button.
    * **Example**: <button accesskey="S">Submit</button> (The user can press "Alt + S" to activate the button).
13. **title**:
    * Provides additional information about the button, which is often displayed as a tooltip when the user hovers over the button.
    * **Example**: <button title="Click to submit">Submit</button>
14. **lang**:
    * Specifies the language of the button’s text content.
    * **Example**: <button lang="en">Submit</button>
15. **style**:
    * Allows you to apply inline CSS styles to the button.
    * **Example**: <button style="color: red; background-color: yellow;">Click Me</button>

**Attributes of the <table> Element**

1. **border**:
   * Specifies the width of the table's border. In HTML5, this is typically handled by CSS, but in HTML4, you could directly set border="1".
   * Example: <table border="1">
2. **cellpadding**:
   * Sets the amount of space (padding) between the cell content and the cell border. This attribute is deprecated in HTML5 and is generally handled by CSS (padding).
   * Example: <table cellpadding="5">
3. **cellspacing**:
   * Defines the space between individual cells in a table. This is also deprecated in HTML5 and usually handled by CSS (border-spacing).
   * Example: <table cellspacing="5">
4. **width**:
   * Specifies the width of the table. It can be set as a percentage (relative to its container) or as an absolute pixel value. CSS is now the preferred way to set width.
   * Example: <table width="80%">
5. **height**:
   * Specifies the height of the table. Like width, it can be a percentage or a fixed value. CSS is preferred for controlling height.
   * Example: <table height="200px">
6. **align**:
   * Aligns the table on the page. Common values are left, right, and center. This attribute is deprecated in HTML5; CSS (text-align or margin) should be used instead.
   * Example: <table align="center">
7. **bgcolor**:
   * Sets the background color for the table. This is an HTML4 attribute and is deprecated in HTML5, with CSS (background-color) preferred.
   * Example: <table bgcolor="lightblue">
8. **summary**:
   * Provides a summary of the table’s content, mainly for accessibility purposes. This is no longer widely supported in modern HTML5 but was used to describe complex tables to assistive technologies.
   * Example: <table summary="This table lists quarterly financial results.">
9. **frame**:
   * Specifies which parts of the table’s border should be visible. Possible values include void, above, below, hsides, lhs, rhs, vsides, box, and border.
   * Example: <table frame="box">
10. **rules**:
    * Determines which lines should be visible between rows, columns, or both. Possible values include none, groups, rows, cols, and all.
    * Example: <table rules="all">

The <tr>, <td>, and <th> elements are essential components of an HTML table:

* **<tr>** defines a row in a table.
* **<td>** represents a cell within a row, typically for regular data.
* **<th>** is similar to <td> but used for header cells, which are usually bold and centered by default.

Here are the attributes for each of these elements.

**Attributes of <tr> (Table Row)**

1. **align**:
   * Aligns the content horizontally within each cell in the row (left, center, right). This attribute is deprecated; CSS (text-align) is recommended.
   * Example: <tr align="center">
2. **bgcolor**:
   * Sets the background color for the row. This attribute is deprecated; use CSS (background-color).
   * Example: <tr bgcolor="#f2f2f2">
3. **valign**:
   * Vertically aligns content within each cell in the row (top, middle, bottom, baseline). This attribute is deprecated; CSS (vertical-align) is recommended.
   * Example: <tr valign="top">

**Example of <tr> with Attributes**

|  |
| --- |
| .  <tr align="center" bgcolor="#f9f9f9" *valign*="top">              <td>Apples</td>              <td>$1.00</td>              <td>10</td>            </tr>  . |

**Attributes of <td> (Table Data Cell)**

1. **colspan**:
   * Merges the cell across multiple columns.
   * Example: <td colspan="2">Merged Cell</td>
2. **rowspan**:
   * Merges the cell across multiple rows.
   * Example: <td rowspan="2">Merged Cell</td>
3. **headers**:
   * Associates the cell with header cells for accessibility, especially useful in complex tables.
   * Example: <td headers="header-id">Data</td>
4. **align**:
   * Aligns the cell content horizontally (left, center, right). This is deprecated; use CSS instead.
   * Example: <td align="right">
5. **valign**:
   * Vertically aligns the content (top, middle, bottom, baseline). This is deprecated; use CSS instead.
   * Example: <td valign="middle">
6. **bgcolor**:
   * Sets the cell background color. Deprecated; use CSS (background-color).
   * Example: <td bgcolor="#e0e0e0">
7. **width** and **height**:
   * Defines the width and height of the cell. This is deprecated; CSS is preferred.
   * Example: <td width="100px" height="50px">Data</td>

**Example of <td> with Attributes**

|  |
| --- |
| .  <tr>     <td *colspan*="2" align="center" bgcolor="#e0e0e0">Total Price</td>     <td>$50</td>  </tr>  . |

**Attributes of <th> (Table Header Cell)**

1. **scope**:
   * Defines the header cell’s scope in relation to its associated cells (row, col, rowgroup, colgroup). Useful for accessibility.
   * Example: <th scope="col">Price</th>
2. **colspan** and **rowspan**:
   * Similar to <td>, allowing the header to span multiple columns or rows.
   * Example: <th colspan="2">Header</th>
3. **headers**:
   * Associates the header with specific cells, helpful in complex tables.
   * Example: <th headers="id123">Category</th>
4. **align** and **valign**:
   * Controls the horizontal and vertical alignment of header content. Deprecated; use CSS.
   * Example: <th align="left">Product</th>
5. **abbr**:
   * Provides an abbreviated form of the header text, useful for accessibility and tooltips.
   * Example: <th abbr="Qty">Quantity</th>

**Example of <th> with Attributes**

|  |
| --- |
| .  <tr>    <th *scope*="col" align="left" *abbr*="Qty">Quantity</th>     <th *scope*="col">Price</th>     <th *scope*="col" *colspan*="2">Product</th>  </tr>  . |