

Forms in HTML

The `<form>` element in HTML is used to create a form that can be submitted to a server for processing. Forms are used to collect input from users, such as text, checkboxes, radio buttons, and file uploads.

Here's an example of a simple HTML form:

```
<form action="/submit-form" method="post">
  <label for="user_id">User ID</label>
  <input type="text" id="uid" name="user_id">

  <label for="email">Email:</label>
  <input type="email" id="emailid" name="email">

  <label for="pass">Password:</label>
  <input type="password" id="pass" name="pass">

  <input type="submit" value="Submit">
</form>
```

The browser renders the form as follows:

User ID Email: Password:

The `<form>` element has two attributes, `action` and `method`. The `action` attribute specifies the URL of the server-side script that will handle the form submission, in this case `/submit-form` and the `method` attribute specifies how the form data should be sent to the server, in this case `"post"`.

Inside the `<form>` element, you can use various input elements like `<input>`, `<textarea>`, `<select>` etc. to create form elements. Each input element must have a `name` attribute, which is used to identify the data when it is sent to the server. Also, it is best practice to use labels for form elements which makes it accessible for screen readers and other assistive technologies.

For example, in the above example, the `<input>` element with a `type` attribute of `"text"` is used to create a text field, where a user can enter their User Id. The `<input>` element with a `type` attribute of `"email"` creates an email field. The element with a `type` of `"password"` creates a password field.

The `<input type="submit">` element is used to create a submit button, which is used to transfer the form data to the server when this button is clicked. The `<input type="date">` is used to generate a date picker. Similarly, an `<input type="url">` is used to take a url input.

Other Form Elements

Text Area

The `<textarea>` element is used in HTML forms to create a multi-line input field where the user can enter multiple lines of text. The `<textarea>` element is a block-level element, which means it takes up the full width of its parent container and creates a new line after it.

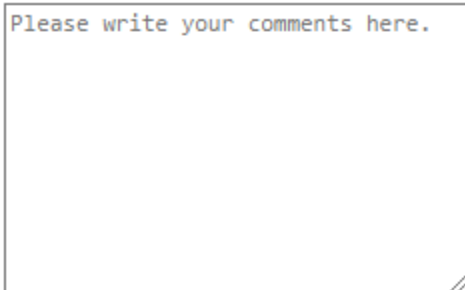
The `<textarea>` element has several attributes that can be used to customize its behavior and appearance, such as:

- `name`: Specifies the name of the textarea, which is used to identify the field when the form is submitted.
- `rows`: Specifies the number of visible text lines in the textarea.
- `cols`: Specifies the visible width of the textarea, in characters.
- `placeholder`: Gives a hint of what should be the expected value of the textarea.
- `readonly`: Specifies that the textarea should be read-only.
- `required`: Specifies that the textarea should be filled out before submitting the form.
- `disabled`: Specifies that the textarea should be disabled and the user cannot interact with it.

You can see the example code below:

```
<textarea name="" id="" cols="30" rows="10" placeholder="Please write your  
comments here."></textarea>
```

The output is rendered by the browser as below:



Radio Buttons

Radio buttons are used in HTML forms to create a group of options where the user can select only one option. They are created using the `<input>` element with the `"type"` attribute set to `"radio"`.

You can see the example code below:

```
<input type="radio" name="diet" value="vegetarian" checked> Vegetarian<br>  
<input type="radio" name="diet" value="non-vegetarian">Non-Vegetarian<br>
```

The name attribute is used to group the radio buttons together, so that only one option can be selected at a time. The value attribute specifies the value that the server will receive when the form is submitted. When the webpage loads, we can make sure that a radio button is selected using the checked attribute.

Check Boxes

In HTML, a checkbox can be created using the `<input>` element with the type attribute set to "checkbox". The `<input>` element should also have a unique id attribute to associate it with a label, and a name attribute to group it with other checkboxes (if desired).

Here is an example of a checkbox in HTML:

```
<label for="checkbox1">Option1</label>
<input type="checkbox" id="checkbox1" name="options">
<label for="checkbox1">Option2</label>
<input type="checkbox" id="checkbox2" name="options">
```

Drop Down List

In HTML, a drop-down list can be created using the `<select>` element. We can use the `<select>` element to create a drop-down list of options, and each option within the list is represented by the `<option>` element.

Here is an example of a basic drop-down list in HTML:

```
<label for="select1">Select an option:</label>
<select id="select1" name="options">
  <option value="opt1">Opt 1</option>
  <option value="opt2">Opt 2</option>
  <option value="opt3">Opt 3</option>
</select>
```

Once the form is submitted, the browser sends the form data to the server as specified by the action and method attributes. It is up to the server-side script to handle the data and do something with it, such as storing it in a database or sending an email.

It is also recommended to validate form data on the client side before submitting it to the server to avoid any errors or bad requests. You can use JavaScript or a library such as jQuery to perform validation of form.

Forms - Method Attribute

The method attribute of the <form> tag is used to specify the HTTP method that should be used when the form is submitted. The two most common values for the method attribute are "GET" and "POST".

The "GET" method is used to retrieve information from the server. When a form is submitted with the "GET" method, the data that the user has entered into the form is appended to the URL as query string parameters. This method is useful when the form is used for searching, or when the form data doesn't need to be kept private.

The "POST" method is used to send information to the server. When a form is submitted with the "POST" method, the data that the user has entered into the form is sent to the server as part of the request body, rather than being appended to the URL. This method is more secure than "GET" and is used when the form data needs to be kept private or when the form is used for updating or creating new data.

In the above image, the method attribute is set to "POST" and the form data will be sent to the URL specified by the action attribute (submit-form).

The Id and Class attributes

In HTML, elements can have an id attribute and a class attribute, which can be used to specify a unique or a group of elements to style with CSS, and to interact with using JavaScript.

The id attribute is used to specify a unique identifier for an element. An element can have only one id attribute and its value must be unique within the entire HTML document. This means that the same id value cannot be given to any other element. The id attribute can be used to style an individual element with CSS or to interact with it using JavaScript.

For example,

```
<p id="unique">This is a unique paragraph</p>
```

The class attribute is used to specify one or more class names for an element, which can be used to group elements together for styling with CSS or interaction with JavaScript. Unlike the id attribute, multiple elements can share the same class name.

For example,

```
<p class="highlight">This is a highlighted paragraph</p>
<p class="highlight">This is another highlighted paragraph</p>
```

HTML5 Attributes

HTML5 is the latest version of HTML, and it introduces a number of new attributes that can be used to improve the functionality and accessibility of web pages. Here are a few examples of commonly used HTML5 attributes:

autofocus: When used on an `<input>` or `<textarea>` element, the autofocus attribute causes the element to automatically receive focus when the page loads, allowing users to begin typing or interacting with the element immediately.

required: When used on an `<input>` element, the required attribute specifies that the input field must be filled out before the form can be submitted.

min, max: When used on an `<input>` element, these attributes specify the minimum and maximum value that the input can have.

formnovalidate: When used on an `<input>` element that is a submit button, this attribute tells the browser that the form should not be validated when the user clicks the button.