Module 2, Day 2

Module 2, Day 2, Lesson 1: HTML Forms

ntroduction

HTML forms are key to interactive websites as they allow users to enter data and interact with the site. They are used everywhere from search boxes to login screens and data entry forms.

HTML Forms

- An HTML form is created using the `<form>` element. The form element acts as a container for all the different input elements like text fields, checkboxes, radio buttons, etc.
- Example:

```
<form>
  <!-- form elements go here -->
```

- `</form>`

Labels

- The `<label>` element is used to add a text description to an input field. This is important for accessibility and for helping users understand what data should be entered into a field.
- Labels are associated with inputs by using the `for` attribute on the label, which matches the `id` of the input.
- Example:

```
<label for="firstname">First Name:</label>
```

- `<input type="text" id="firstname" name="firstname">`

nputs

- The `<input>` element is used to create different types of input fields, like text fields, checkboxes, radio buttons, submit buttons, etc.
- The `type` attribute of the input tag defines what type of input field to display.

Module 2, Day 2

- The most common type of input. It allows the user to type in a single line of text.
- Example:
- `<input type="text" id="lastname" name="lastname">`

Radio Buttons

- Radio buttons let the user select one option from a predefined set.
- Example:

```
<input type="radio" id="gender" name="gender" value="male">
```

- `<input type="radio" id="gender" name="gender" value="female">`

Checkboxes

- Checkboxes allow the user to select one or more options from a set.
- Example:
- `<input type="checkbox" id="terms" name="terms" value="agree">`

Buttons

- The `<button>` tag defines a clickable button.
- By default, a button submits the form data, but this can be overridden by setting the `type` attribute to `button`.
- Example:
- `<button type="submit">Submit</button>`

Dropdown Menus

- The `<select>` element is used to create a dropdown list. The `<option>` elements inside the select define the available options in the list.
- Example:

Module 2, Day 2

- `</select>`

Form Attributes: Action, Target, Methods, GET & POST

Action Attribute

- The `action` attribute defines where the form's data should be sent when the form is submitted.
- The value can be an absolute or relative URL.
- If this attribute is not specified, the form data is sent to the URL of the page containing the form.
- Example:
- `<form action="/action_page.php">`

Target Attribute

- The `target` attribute specifies where to display the response after submitting the form.
- The attribute can have values like `_blank` (the response is displayed in a new window or tab) or `_self` (it's displayed in the same frame).
- Example:
- `<form action="/action_page.php" target="_blank">`

Method Attribute: GET & POST

- The `method` attribute specifies the HTTP method to use when sending the form data.
- `GET` appends the form data to the URL in the form of a query string. The data is visible in the browser's address bar, and has limits on the amount of data to send. However, it is useful for non-sensitive data, like query strings in Google.
- `POST` sends the form data as HTTP post transaction. The data is included in the body of the form and is not visible in the URL. It has no size limitations.
- Example:

Module 2, Day 2

More details on the attributes `type`, `id`, `name`, and `for`.

`type` Attribute:

The `type` attribute in an `<input>` element declares what kind of input control to display such as text fields, checkboxes, radio buttons, submit buttons, etc. The type attribute can take several values, including:

- `text`: Defines a one-line text input field
- `submit`: Defines a submit button (for form submission)
- `radio`: Defines a radio button (for mutually exclusive choices)
- `checkbox`: Defines a checkbox (for multiple selectable options)
- `password`: Defines a password field
- `button`: Defines a clickable button
- `number`: Defines a field for numeric input
- `date`: Defines a date control
- `email`: Defines a field for email input
- `file`: Defines a file-select field and a "Browse" button (for file uploads)

Example:

```
<input type="text" id="name" name="name">
```

`id` Attribute:

The `id` attribute specifies a unique id for an HTML element. The id value can be used in CSS and JavaScript to perform certain tasks for the element with the specific id.

In forms, the `id` attribute is often used to associate `<label>` elements with `<input>` elements, which improves accessibility.

Example:

```
<label for="name">Name:</label>
<input type="text" id="name" name="name">
```

Module 2, Day 2

The `name` attribute in `<input>` elements specifies the name for the input element. The name attribute is used to reference form data after a form is submitted, or to reference the element in a JavaScript.

Example:

```
<input type="text" id="name" name="name">
```

`for` Attribute:

The `for` attribute in `<label>` elements associates the label with an `<input>` element. The value of the `for` attribute must be equal to the id attribute of the `<input>` element. This is especially useful for users who have difficulty clicking, as they can click on the label to select the input element.

Example:

```
<label for="name">Name:</label>
<input type="text" id="name" name="name">
```

This association between the label and the corresponding input field improves form accessibility, as clicking the label will focus/activate the associated input field. Screen readers will also read out the label when the user is focused on the input field.

Module 2, Day 2, Lesson 2: Designing an Effective Form

Lesson 2: Review of Form Elements

- `<form>`: This tag acts as a container for all the form controls like text fields, checkboxes, radio buttons, dropdown lists, etc.

- `<input>`: This is the most commonly used form element. By changing the `type` attribute, you can create many different kinds of input elements, such as text fields (`type="text"`), checkboxes (`type="checkbox"`), and radio buttons (`type="radio"`).

- `<label>`: This element represents a caption for an item in a user interface and improves accessibility. It should be associated with an input element by using the `for` attribute.

- `<button>`: This tag creates a clickable button. By default, it submits form data, but this can be overridden by setting the `type` attribute to `button`.

- `<select>` and `<option>`: These tags are used to create a dropdown list. The `<option>` elements inside the select define the available options in the list.

- `<textarea>`: This tag defines a multi-line text input control.

Module 2, Day 2

When designing a form, it's not just about understanding the code - it's also about creating a good user experience. Here are some best practices:

1. Keep it Simple: A user-friendly form should be simple and intuitive. Only ask for the information you really need.
2. Group Related Fields: Related fields should be grouped together to help users understand what is required of them at a glance.
3. Use Labels: Always use labels with your form inputs to ensure users know what each field is for. This is also important for accessibility.
4. Use Appropriate Input Types: Use the appropriate input types to make it easier for users to input data. For example, use checkboxes for multiple options, radio buttons for exclusive options, and dropdowns for long lists of options.
5. Make Use of Button Text: The button should make it clear what will happen when it's clicked. For example, "Submit", "Sign Up", or "Join Now" are all clear button texts.
6. Ensure Accessibility: Your form should be accessible to all users, including those using assistive technologies. This includes proper use of `<label>` elements, providing alternative text, and ensuring your form is navigable via keyboard.

Codepen Resources

Lesson 1:
Updated form example: https://codepen.io/shafferma08/pen/zYMmbpw

Lesson 2:
Code-along example: https://codepen.io/shafferma08/pen/ZEmqPoB
Table/Form example: https://codepen.io/shafferma08/pen/poQxYOb

Web Accessibility Resources

Web Content Accessibility Guidelines (WCAG): The WCAG is a set of recommendations for making web content more accessible to people with disabilities.

W3C's Web Accessibility Initiative (WAI): WAI provides strategies, guidelines, and resources to help make the web accessible to people with disabilities.

WebAIM (Web Accessibility In Mind): WebAIM offers a wide range of resources, including training, technical assistance, tools, and guidelines for creating accessible web content.

A11Y Project: The A11Y Project is a community-driven effort to make web accessibility easier. It provides a checklist, resources, and patterns for building accessible web components.

Mozilla Accessibility: Mozilla Developer Network (MDN) provides a comprehensive accessibility section with explanations, guides, and best practices.

Module 2, Day 2