December 8, 2021

Creating Forms in HTML

Course Overview:

-HTML form basics

-Various from input types

-Organizing form inputs

-styling and validating forms

**Understanding HTML Forms**

* Understanding HTML Forms
* Understanding form components

-HTML presents server data to users.

-HTML forms allow users to provide data to server.

**-form element>>input elements>>submit button**

* Building and submitting forms
* Demo: Setting Up Your editor

-In stead of using live server, you can do a .vscode folder and put a “launch.json” file in it. Then in your package.json file,

You need these script tags:

Text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

Text

Description automatically generated

Then tab on the left side, run the code without debugging.

* Demo: Creating the First Form

-go to Aside and add form

* Summary

**Using Form Inputs**

* **Using Form Inputs**

-single line text(numbers or dates, multiple line text(paragraphs or comments), selection(drop down, lists, sliders)

-**basic single line text inputs**

**-text/search, hidden, password**

* Demo: Texts Inputs

-open contact.html(02/demo/begin), add form in here

* **Additional Data Types**

-email, url, telephone, number, date, time

-custom keyboards

-**automatic drop down menu for date and time when you select it as input type**

-**input types for numbers** you can set a min and max: min=”1” max = “10”, and on the browser it will prompt user that they can’t go above 10 and it won’t let you select below 1 or go above 10

-able to limit what users can input by doing max, min and validation

* Selection Inputs

-**select(list), radio buttons, check box, range**

Demo: Select List

-Changed severity to severityRange and we get a slider, the **min and max** lets you now how big of a range of severity your user selected

-Different browsers use different types of sliders and controls, that’s why we must test on different browsers

* Demo: Checkboxes and Radio Buttons

-Feedback type: we put **<option value>**

Text

Description automatically generated

-All these are choices we give the users, and in the browser they get a drop down menu of those options.

**-<optgroup label=”store”>** The stores in the browser will have their own list and the other choices will not be indented. Store will be a header with the list of options under it.

-allow to pick multiple options with **‘multiple = “multiple**” size= 6’ this allows the user to pick up to 6 choices, just have to hold down the control button to select more than one option from the browser side.

Graphical user interface, text, website

Description automatically generated

**Checkbox, radio buttons:** if you name more than thing that specific name, the user can only select one from it.

Text

Description automatically generated

To see this in Dev tools: go to inspect>>Network>>Name>>Headers>>Payload

There you’ll see the form & input values

* **Summary**

-Data types help usability and accuracy

-Different input types help clarify request, UX best experience

**Organizing Form Inputs**

**-Using Fieldsets**

Table

Description automatically generated

-organizing your inputs, visually groping elements together. Contact information and group multiple sets of information into a logical collection.

-add a fieldset to form, enclose the contact information with <fieldset>, also add a <legend>, the text that appears that the top of our fieldset

-Demo, grouped the feedback fieldset as well

**-Applying Labels**

-associate content with specific input, label makes it more semantically important

**-Labels**

-**Explicit reference by id:**

<label for=”firstName”>First Name:</label>

<input type=”text” id=”firstName” />

-**Explicit reference by enclosing**

<label>First Name:

<input type=”text” id=”firstName” />

</label>

The label is now associated with the input, so if you click on the label, the cursor moves into that input box

-doing this also useful for people using assistive technology or vision problems

-**Navigating with Tabs(how nav occurs)**

**-Tab order**

-Default is document order

-Use TabIndex attribute to specify

-specifying zero moves to the end

-“tabindex=”#” -that’s the order the tab will go in, if you set the first input to tabindex=”1”, then the first tab of the page will go to input 1

**\*\*When using the “tabindex”, make sure that you use it throughout your entire page and your form because the tab will follow throughout your page that doesn’t even relate to your form.\*\***

**Summary:**

**Organizing makes forms more usable**

**Consider different input types/tools/adaptative technology**

**Styling and Validating Forms**

* **Styling and Validating Forms**

-**using CSS selectors**

-Apply styles to form elements

-input

-select

-text area

-fieldset/legend

-label

-button

-The way to apply styles is to apply it to specific types

Graphical user interface, text, application

Description automatically generated

-apply styles to multiple types, use commas to separate those selectors allow you to apply the same set of styles to multiple types or multiple elements inside your form.

-**Validating form inputs on the client**

-**Using styles to reflect validity**

* **Demo: Applying Styles(folder 4/begin/contact.html/sytles.css)**
* -Styles.css>> fieldset{
* border-color:#a593c2;
* border-width: 3px;
* border-style: solid;
* }
* fieldset>legend{
* color:#a593c2
* } -<<this is calling the legend fieldset since there’s 2
* -for labels inside a paragraph, you can do display block:
* p>label {
* display: block;
* }
* -for input stylings on the demo: It made the input text area rounded
* Input, textarea {
* border-radius: 7px;
* }
* -if you wanted to do styling for only certain “type” this is the method:
* Text

  Description automatically generated

* **Validating Client Input**

**Text, application

Description automatically generated**

* **Demo: Using Validation Attributes**

-back in contact.html, we want to get name and feedback, and go put “required”. If you try to submit the form without the required being filled, the browser will alert you and tell you to fill it out

-now go back and require minlength=”10” and maxlength=”75” to name

-use pattern for phone number:pattern=

**Text

Description automatically generated**

* **Applying Styles Based on Validity**
* **Graphical user interface, application

  Description automatically generated**
* Input:valid{
* Border-color: green;
* }
* Input:invalid {
* Border-color: red;
* }
* **And while a user is filling it out, we don’t want the border to have a color:**
* input:focus:invalid{
* border-color: black;
* }
* **When styling with an attribute:**
* input:required,

textarea:required {

* border-style: dashed;
* }
* **Summary**
* **-using styles for customization**
* **Validation on the client is helpful**
* **Validation on the server is required**
* **Styling can include validity of inputs of what we need on feedbacks**