



Forms and Their Styling

Lesson 6



Lesson plan

1

HTML Form Elements

2

Styling Form Elements

3

Form Validation

4

Other Interactive Tags

Registration

Full Name
E.g. John Smith

Username
johnWC98

Email
johnsmith@hotmail.com

Phone Number
012-345-6789

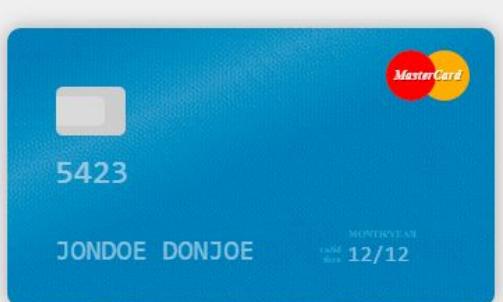
Password

Confirm Password

Gender

Male Female Prefer not to say

Register



Start typing in here to take over and try it out

5423 Jondoe Donjoe

12 / 12

121

Submit

Send us a Message

CodingNepal

Enter your phone

Write your message

Send Message

Sending your message...

Price Range

Use slider or enter min and max price

Min

2500

- Max

7500

Subscribe to the newsletter

your@email.com

Subscribe

INGREDIENTS

Beef

Pork

Chicken

Vegetables

Start Date

04-05-2020

End Date

22-05-2020

May 2020

Su	Mo	Tu	We	Th	Fr	Sa
18	26	27	28	29	30	1
19	3	4	5	6	7	8
20	10	11	12	13	14	15
21	17	18	19	20	21	22
22	24	25	26	27	28	29
23	31	1	2	3	4	5

Clear

NUTRITION

3

Gluten Free

Your email address:

test@example.org

Interactive form demo

Test message *

Optional files

Browse...

No files selected.

Send to me

Demo source code

```
<!-- source HTML for the demo -->
<form
  action="https://api.mailslurp.com/forms"
  method="post"
  enctype="multipart/form-data">

  <!-- destination specified with hidden '_to' field -->
  <input
    name="_to"
    type="hidden"
    value="test@example.org">

  <!-- use any named inputs, selects, textareas etc -->
  <label>Test message</label>
  <textarea name="message"></textarea>

  <!-- files will be sent as attachments -->
  <label>Optional files</label>
  <input multiple name="files" type="file">

  <button type="submit">Send to me</button>

</form>
```

<form> Attributes



action – The URL where the form data will be sent

method – The HTTP method used for sending form data

GET – Used to retrieve data

POST – Used to send data, especially sensitive information

```
<form action="submit.php" method="post">
```

<input>



- Used to create interactive form controls for user data input.
- The default type is **text**
- One of the most powerful and complex HTML elements due to the vast number of input types and attributes.

<input type="text">



input attributes

- customize `<input>`

- These attributes add features and behaviors to the `<input>` elements.
- Attribute examples include **size**, **value**, **maxlength**, **required**, and many more.

```
<input  
    type="text"  
    name="bro"  
    required  
    minlength="2"  
    maxlength="8"  
/>
```

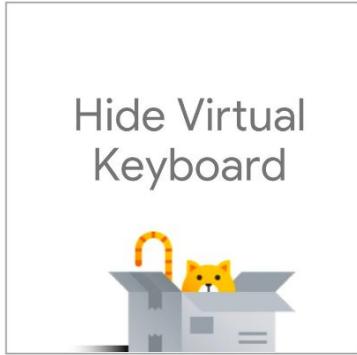
input types

Why is it important to use the proper type?

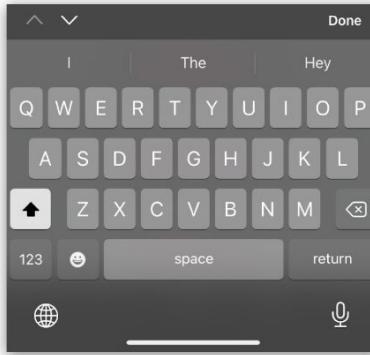
- User Experience
- Validation
- Accessibility
- Styling and Behavior
- Improved Semantics

<input type="button">	Button
<input type="checkbox">	<input checked="" type="checkbox"/>
<input type="color">	<input type="color"/>
<input type="date">	dd-mm-yyyy 
<input type="email">	faracz60@gmail.com
<input type="file">	Choose File <input type="file"/> No file chosen
<input type="hidden">	
<input type="image">	
<input type="number">	5 
<input type="password">	*****
<input type="radio">	<input checked="" type="radio"/> <input type="radio"/>
<input type="range">	<input type="range"/>
<input type="reset">	Reset
<input type="submit">	Submit
<input type="text">	codewithfaraz
<input type="url">	https://www.codewithfaraz.com

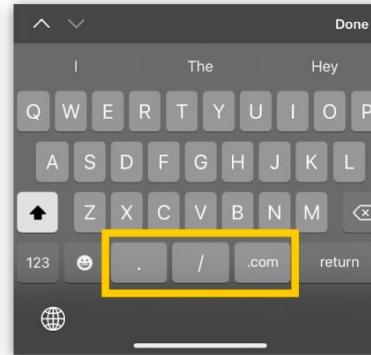
<input inputmode="..">



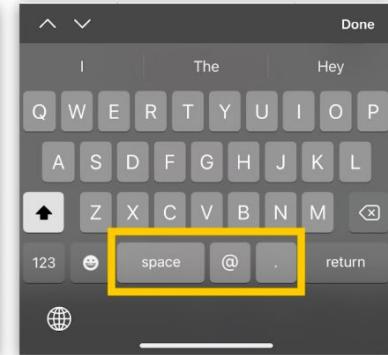
"none"



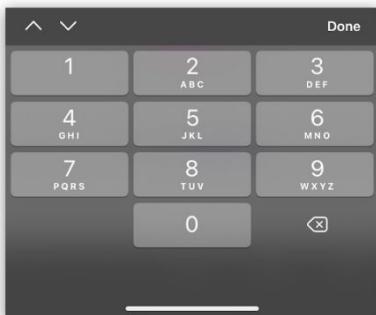
"text"



"url"



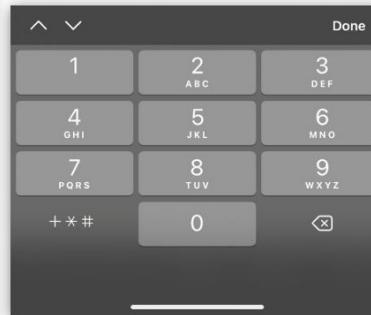
"email"



"numeric"



"search"



"tel"



"decimal"

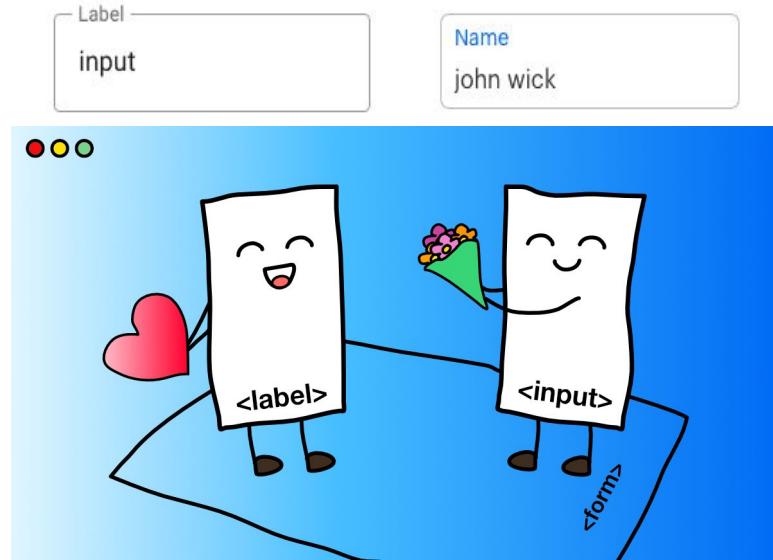
label

- description for a form element

- linked using the **id** attribute
- clicking on the **label** **focuses** on the associated form element
- ! every form element should have a corresponding label
- can be used for styling elements that are otherwise difficult to style

```
<input type="radio" name="fruit" value="apple" id="apple">  
<label for="apple"> Apple </label>
```

```
<label>  
  Apple  
  <input type="radio" name="fruit" value="apple">  
</label>
```



Checkbox Usage Rules

- used to let a user select **one or more** options of a limited number of choices.

```
<input type="checkbox" name="vegetables" value="tomato"  
class="input-checkbox" id="checkbox-tomato">
```

Required Attributes:

value – Must be unique for each checkbox



Tomato



Onion

Optional:

name – Should be the same for all checkboxes in a group



Lettuce



Capcicum

Radio Button Usage Rules

- used to let a user select **one** option of a limited number of choices.

```
<input type="radio" name="vegetables" value="onion" class="radio visually-hidden" id="vegetables-onion">
```



Required Attributes:

name – Required Attributes

value – Must be unique for each radio button

textarea

- Allows multi-line text input
- By default, it is resizable. This can be disabled using CSS



textarea

Attributes:

cols – specifies the width of the textarea in terms of the number of text columns

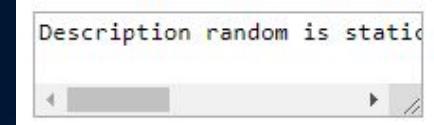
rows – specifies the number of text lines it can display at a time (height)

instead of using this attributes, you might set the width and height in your CSS

if you want to prevent users from resizing:

```
textarea {  
    resize: none;  
}
```

```
<div class="form-row">  
    <textarea name="basic-textarea" id="basic-textarea"  
        class="form-control" cols="30" rows="10"></textarea>  
    <label for="basic-textarea">Basic Textarea</label>  
</div>
```



form validation

- Validates form data according to specific rules

- Methods for validation include:

- **html** →
- **js**

1. Use the appropriate **input type**
2. Add the **required** attribute where necessary
3. Use the **pattern** attribute for custom validation rules
4. Style inputs and validation messages using pseudo-classes like **:invalid**, **:valid**

The screenshot shows a sign-up form with several validation errors:

- A text input field containing "test" has a red border and a validation message "Last name" below it.
- An email input field containing "test@test.com" has a red border and a validation message "Please fill out this field." below it.
- A password input field containing "....." has a red border and a validation message "An account has already been registered with this email. Log in to your account." below it.
- A green "Sign Up" button is at the bottom.

The screenshot shows a sign-up form with validation errors:

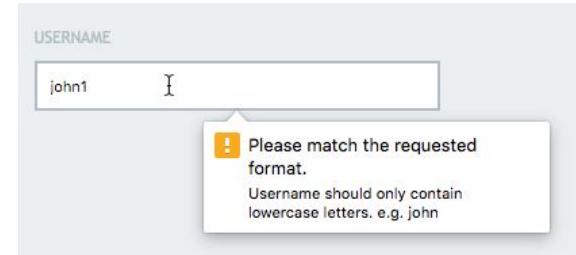
- Username input field: "js" is highlighted in red with the error message "Username must be between 3 and 25 characters".
- Email input field: "hello@example.com" is highlighted in green.
- Password input field: "..." is highlighted in red with the error message "Password must has at least 8 characters that include at least 1 lowercase character, 1 uppercase characters, 1 number, and 1 special character in (!@#\$%^&*)".
- Confirm Password input field: "Reenter your password" is highlighted in red with the error message "Please enter the password again".
- A blue "SIGN UP" button is at the bottom.

input attribute pattern

- helps ensure data is entered in the correct format
- provides a prompt or error message if the data doesn't match the pattern
- ask ChatGPT to help you create the right pattern

pattern=" [A-Za-z]{5}"

This pattern requires 5 characters, allowing both uppercase and lowercase Latin letters.



phone +38 (____) ___ - ___

phone +1 (111)11-111-111|

phone +44 (____) ___ - ___

phone +7(____) ___ - ___

```
<input type="email" pattern=" [a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}\$">
```

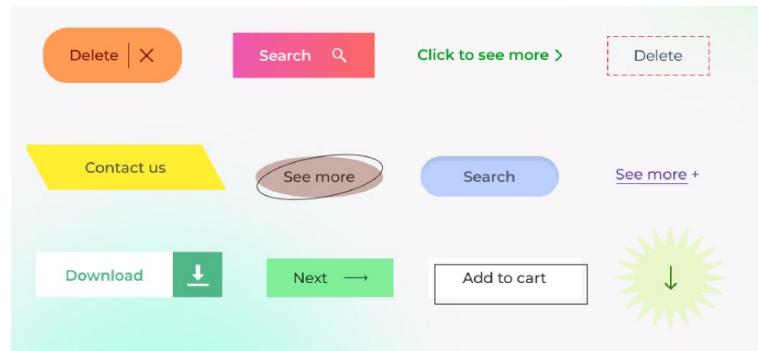
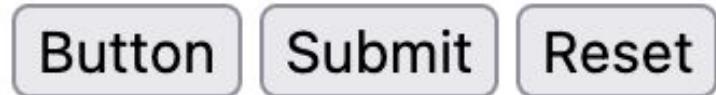
button

button – A generic button for any action

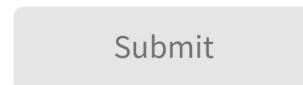
submit – Submits the form data to the server

reset – Resets the form data to its initial state

The **disabled** attribute can be used to prevent the form from being submitted



Button Enabled



Button Disabled

(-) 2 (+)

What is your rate?

★ ★ ★ ★ ★

Please share your opinion about the product



Drop Us A Message

Name Email Address

Write Us A Message

Send

Captcha: N H T T 1 8 7

Reset Enter Captcha

Sign Up Sign In

Email

Username

Password

SIGN UP

Form

Pizza Pasta

Cookies

I love chocolate

Submit Reset form

00 : 00 : 00 : 000

Pause Start Reset

Enter your mail

Subscribe

<input type="checkbox"/>	Checks
<input type="radio"/>	Checked
<input disabled="" type="radio"/>	Disabled
<input checked="" disabled="" type="radio"/>	Checked + Disabled

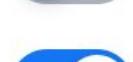
<input type="checkbox"/>	Checks
<input checked="" type="checkbox"/>	Checked

<input disabled="" type="checkbox"/>	Disabled
<input checked="" disabled="" type="checkbox"/>	Checked + Disabled

Toggle button



Setting one



Setting two



Disabled off



Disabled on



<input>

radio & checkbox

- radio and checkbox elements cannot be easily styled directly
(though it is possible)
- Typically, the **input** element is hidden, and the associated **label** is styled instead
- Always consider the **checked** and **disabled** states for styling

```
<input type="checkbox" name="name1" value="value1">
<input type="checkbox" name="name2" value="value2" checked>
<input type="radio" name="name" value="value01">
<input type="radio" name="name" value="value02" checked>
```

Styling Radio & Checkbox

```
<input type="radio" name="weight" value="weight0" class="radio visually-hidden" id="weight0">  
<label for="weight0">0%</label>
```

Styles for Unchecked Elements:

```
.radio:not( :checked) + label {}
```



Styles for Checked Elements:

```
.radio:checked + label {}
```

Accessible input hiding

```
<input type="radio" class="visually-hidden"> display: none;  
  
.visually-hidden:not(:focus):not(:active) {  
  clip: rect(0 0 0 0);  
  clip-path: inset(50%);  
  height: 1px;  
  overflow: hidden;  
  position: absolute;  
  white-space: nowrap;  
  width: 1px;  
}  
}
```

appearance

appearance: none – resets the appearance of an element to a consistent style across all browsers and operating systems

appearance other than none – used to apply specific styles to elements that do not have those styles by default

appearance: searchfield;

The default styling applied by the browser:

appearance: auto;

search:

text:

date: dd.mm.yyyy, --:--

radio:

checkbox:

search:

text:

date: dd.mm.yyyy, --:--

radio:

checkbox:

User interface **pseudo-class** selectors

- **:disabled** – element that is disabled (using the `disabled` attribute)

Default

A screenshot of a web form showing a text input field. The label "Label here *" is positioned above the input field. The input field itself contains the placeholder text "Form text here".

- **:enabled** – elements that do not have the `disabled` attribute

Disabled

A screenshot of a web form showing a text input field. The label "Label here *" is positioned above the input field. The input field itself contains the placeholder text "Form text here". The entire input field is highlighted with a light gray background, indicating it is disabled.

- **:default** – the element that is the `default` among a group of similar elements

User interface **pseudo-class** selectors

- **:checked** – radio or checkbox elements that have the **checked** attribute or have been **selected by the user**

demo

- **:indeterminate** – radio or checkbox elements that are in an intermediate state, **without** the checked attribute or **not selected by the user**

demo

Indeterminate Checkboxes

	Superheroes
<input checked="" type="checkbox"/>	Captain Marvel
<input type="checkbox"/>	Thor
<input type="checkbox"/>	Iron Man
<input type="checkbox"/>	Spider-Man

User interface **pseudo-class** selectors

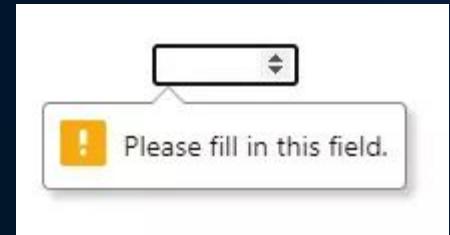
- **:in-range** – highlights an element when the user's input falls within the **specified range**
-

- **:out-of-range** – highlights an element when the user's input falls **outside** the specified range
-

- **input:in-range { }**
<input min="1" max="10">

User interface **pseudo-class** selectors

- **:required** – targets elements that have the **required attribute** (indicating that the field is mandatory)
-



- **:optional** – targets elements that **do not have** the required attribute
-

User interface **pseudo-class** selectors

- **:read-only** – targets elements with the **readonly** attribute (indicating that the user cannot change the element's value)
 - **:read-write** – targets elements that **do not have** the **readonly** attribute
-

Read-Only Input

This can only be copied



User interface **pseudo-class** selectors

- **:target** – applies styles to elements that are referenced by a fragment identifier (ID) in a URL.

```
<a href="#one">Link</a>  
<p id="one"> Target element </p>  
  
p:target {  
    background-color: red;  
    color: white;  
}
```

[Change 1st line.](#)

testing the target selector

User interface **pseudo-class** selectors

- **:valid** – highlights an element when the entered data meets the specified patterns
- **:invalid** – highlights an element when the entered data does not meet the specified patterns.

The image shows two rectangular boxes representing user interface elements. The left box is labeled "Valid" and contains a label "Label *" above a white input field with a thin gray border, containing the placeholder "Input text". The right box is labeled "Invalid" and contains a similar setup, but the input field has a red border around its entire perimeter, indicating it is invalid.

styling valid & invalid

Data entered correctly:

```
input:valid {border-color: green;}
```

valid

Data entered incorrectly:

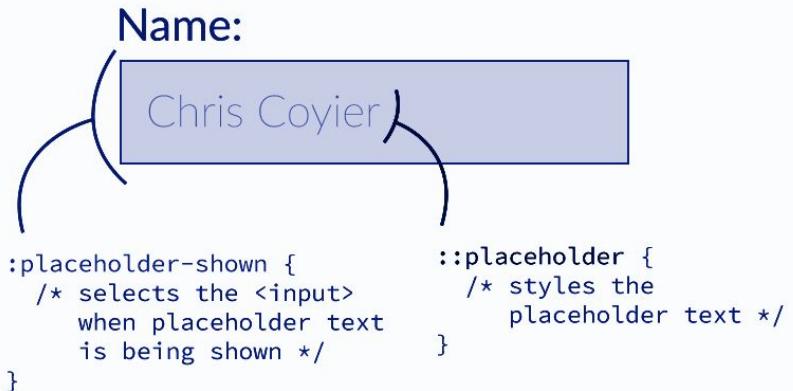
```
input:invalid {border-color: red;}
```

input

Note: This works even if the input is empty, because an empty value is also considered invalid.

Pseudo-elements

::placeholder – Styles the placeholder text inside an **input** or **textarea**, when nothing is entered into it



Pseudo-class

:placeholder-shown – represents any **input** or **textarea** element that is currently displaying placeholder text.

styling valid & invalid

Considering whether data has been entered into the input:

```
input:invalid:not(:placeholder-shown) {border-color: red;}  
input:valid:not(:placeholder-shown) {border-color: green;}
```

This checks whether the placeholder is visible (it's not visible if data has been entered in the field)



styling valid & invalid

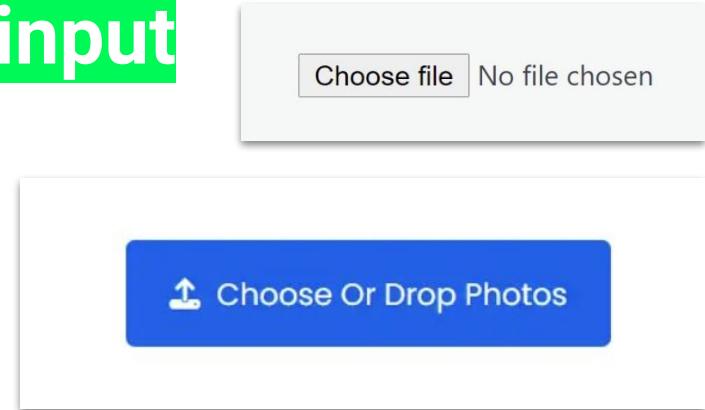
Considering whether the input is focused:

```
input:invalid:not( :placeholder-shown ) :not( :focus )  
{border-color: red}
```

This checks whether the placeholder is visible and whether the field is in focus. If the field is focused, the user might still be entering data, so it may be too early to validate.

styling file upload input

- Styled using pseudo-elements.
- Icons can be added using pseudo-element or by adding wrapper for input



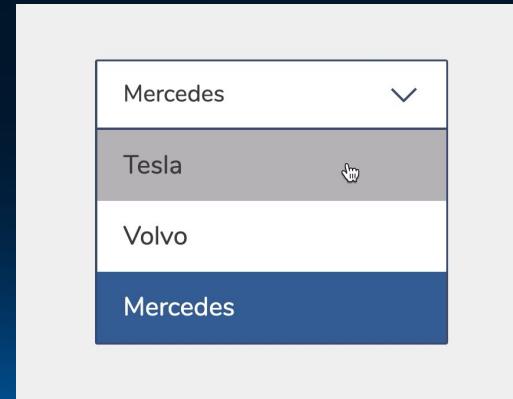
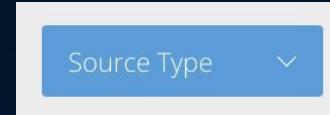
::file-selector-button – Styles the button for `<input type="file">`

`input[type="file"]::file-selector-button {}`

`multiple` - attribute allows to add more than one file
`accept` - attribute allows to specify acceptable file types

select

- Dropdown menu allowing the selection of one or multiple options
- The dropdown button can be partially styled
- The dropdown cannot be natively styled



select

```
<select name="fruit">
  <option value="apple">Apple</option>
  <option value="orange">Orange</option>
  <option value="berry">Berry</option>
</select>
```

option – an element in a dropdown list

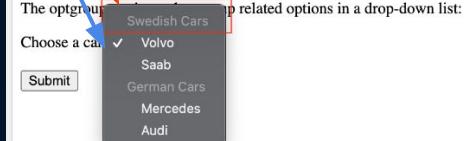
value – the value of the element that will be sent to the server; a required attribute

optgroup – an element used to group items in a dropdown list

label – an attribute that sets the title for a group (the title is not clickable)

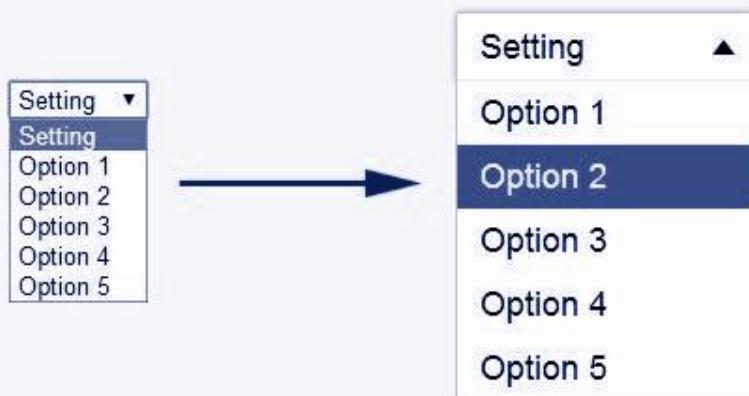
```
<form action="/action_page.php">
  <label for="cars">Choose a car:</label>
  <select name="cars" id="cars">
    <optgroup label="Swedish Cars">
      <option value="volvo">Volvo</option>
      <option value="saab">Saab</option>
    </optgroup>
    <optgroup label="German Cars">
      <option value="mercedes">Mercedes</option>
      <option value="audi">Audi</option>
    </optgroup>
  </select>
  <br><br>
  <input type="submit" value="Submit">
</form>
```

The optgroup element



styling `select` [A]

Using JavaScript libraries.



They create a separate element next to the `<select>` for styling purposes, while hiding the original `<select>` element.

When an option is selected, the choice is passed to the hidden `<select>`, which is then submitted with the form.

For example, [jQuery Nice Select](#).

[demo](#)

Grouping Elements

- **<fieldset>** – Groups related elements within a form
- **<legend>** – A caption for the fieldset
 - The caption is offset upwards, which can make it difficult to align with other elements
 - It does not respond to **flex** layout properties applied to the **fieldset**

The image shows a screenshot of a web browser displaying a form. At the top, there is a legend with the text "Employee Designation". Below the legend is a fieldset containing two input text fields labeled "Input text", "Login Id:", and "Password:". Underneath the fieldset is a list of four checkbox options: "Software Engineer" (checked), "Data Analyst", "Web Developer", and "Senior Analyst". At the bottom of the form are two buttons: "press" and "Reset". Two blue arrows point from the text "Groups related elements within a form" and "A caption for the fieldset" to the legend and the fieldset respectively.

Funkasic



EVENT Registration

Friday 15th & Saturday 16th February @ 8pm

YOUR DETAILS

Name	Country
Phone	State
Email	Mem No.

TICKETS

Date	<input type="checkbox"/> Fri	<input type="checkbox"/> Sat	Delivery	<input type="checkbox"/> Email	<input type="checkbox"/> Collect
Quantity	<input type="text"/>				
Promotional Code <input type="text"/>					

Deadline for Registration is Friday 4th January

eTickets will be issued to your email address two weeks before the event.
Gates will open three hours before the event gets started.
Please remember to bring identification if collecting your ticket from our events desk.

LET'S GET THIS PARTY STARTED



SUBMIT

User settings

Lore ipsum dolor sit amet, consectetur.

Personal info

First Name	Last Name
Country <input type="text"/>	

Notifications

Weekly reports	<input checked="" type="checkbox"/>
Pull requests	<input checked="" type="checkbox"/>
Deployment triggers	<input checked="" type="checkbox"/>

Security

Run security check upon log in	<input checked="" type="checkbox"/>	
Less	<input type="range"/>	More

Save **Cancel**

Tell us about your home.

Address YOUR INFORMATION IS PRIVATE

Address	
Enter an address	
Address 2	
Apt Number	
City	State
City	State
Zip Code	Zip Code

Bed & Bath WHAT IS YOUR HOME LIKE?

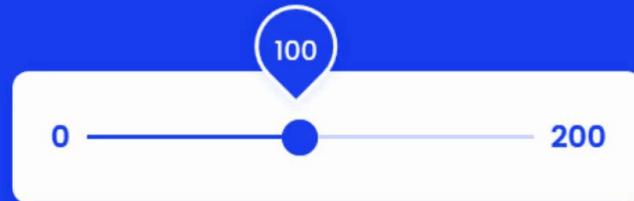
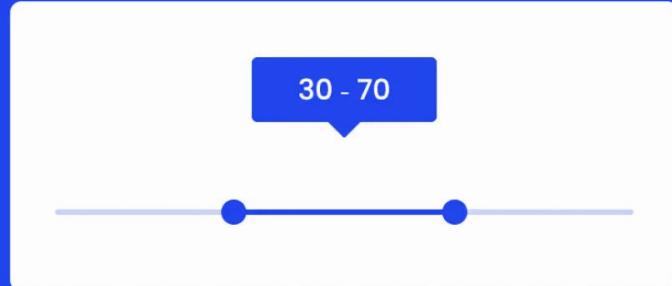
Home Type	Room Type
House	Entire home/apt
Bedrooms	Bathrooms
1 Bedroom	1 Bathroom
King	Queen
0	0
Full	Twin
0	0

Sign me up!

input range

- Has a single slider
- Difficult to style
- usually requires a plugin for more complex controls

For styling, consider using a JS plugin like [AlRangeSlider](#).



input range in HTML

```
<div class="range">
    <button class="range-button is-min"></button>
    <button class="range-button is-max"></button>
    <div class="range-line"></div>
    <div class="range-line is-selected"></div>
    <div class="range-container">
        <div class="range-min"> Min
            <input type="number">
        </div>
        <div class="range-max"> Max
            <input type="number">
        </div>
    </div>
</div>
```

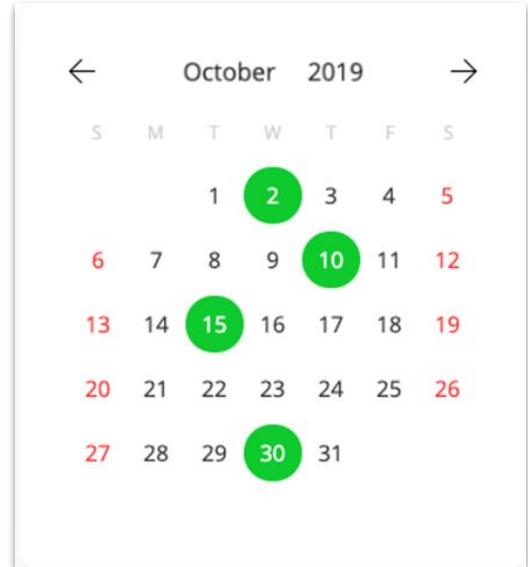
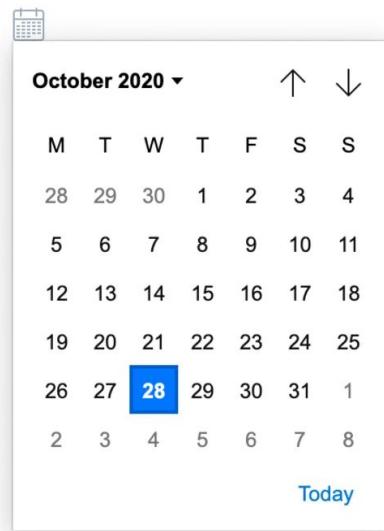
A horizontal slider input with two blue circular handles at positions 300 and 700. Below the slider are two input fields labeled "Min" and "Max" with values "300" and "700" respectively.

styling datepicker

```
<input type="date">
```

- Cannot be styled natively.

- Use JS plugins like [Air datepicker](#) for custom styling



datalist

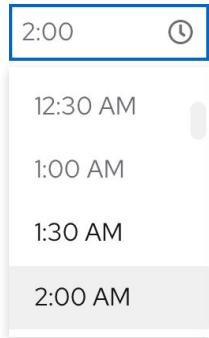
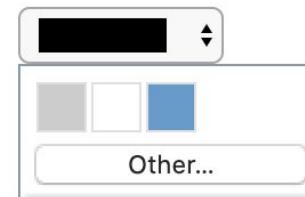
- Provides a list of options for an **input** element.
- You can select an option from the list instead of entering it manually
- Works with specific input types such as:

text, search, range, color, number, date, time, tel, email

The image shows a user interface example. On the left, there is a horizontal slider with a blue bar and a black dot, labeled "FLUENCY". On the right, there is a dropdown menu with a dashed border. The label "List of Books" is above the dropdown. Inside the dropdown, there is a search input field containing the letter "a". Below the input field, a list of book titles is displayed: "Advanced Composite Materials", "Asp.Net 4 Blue Book", and "ADOBE Premiere".

datalist

```
<input list="options" name="input_name">  
<datalist id="options">  
  <option value="option 1">  
  <option value="option 2">  
  <option value="option 3">  
</datalist>
```



progress



– displays the progress of long-running processes.

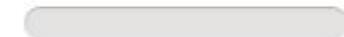
- Progress of filling out a registration form.
- Task completion progress.
- Progress of uploading a large file.
- Progress of course completion in online learning.
- Game progress as levels are completed.



progress

```
<label for="progress-bar">Downloading</label>
<progress
  id="progress-bar"
  value="57"
  max="100"
  title="percentage">
</progress>
```

Progress Bar - 0%



Progress Bar - 100%



Progress Bar - 57%



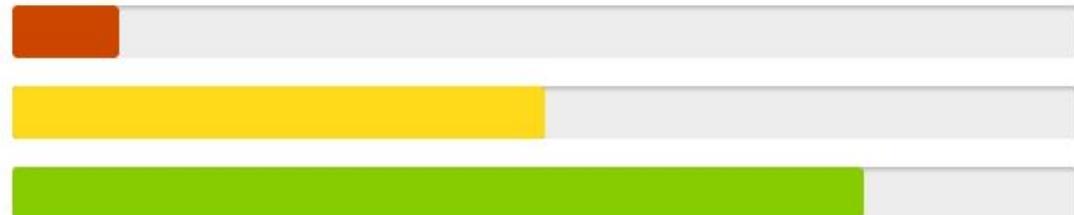
max – the maximum value (a number greater than 0)

value – the current value (a number between 0 and **max**)

meter

– displays a scalar measurement within a known range

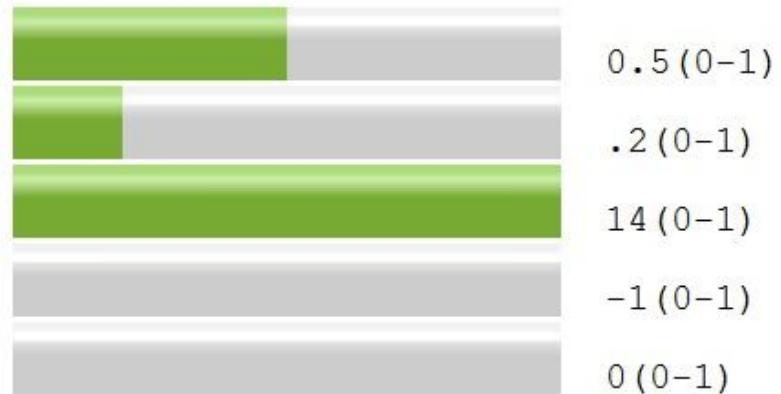
- Password strength
- Battery level
- Sound volume



meter

```
<label for="vote"></label>  
<meter  
    id='vote'  
    value='75'  
    min='0'  
    max='100'>  
</meter>
```

value – current value on the scale
min – minimum value
max – maximum value
low – lower threshold value
high – upper threshold value
optimum – optimal threshold value



styling progress

```
progress {  
    appearance: none;  
}
```

```
progress::-webkit-progress-bar {  
    background-color: grey;  
}
```

```
progress::-webkit-progress-value {  
    background-color: green;  
}
```



91 / 100

meter

```
meter {  
    appearance: none;  
}  
  
meter::-webkit-meter-bar {  
    background: none;  
    /* Required to get rid of the default background property */  
    background-color: grey;  
}  
  
meter::-webkit-meter-optimum-value {  
    background-color: green;  
}
```

Safari



Chrome



Firefox



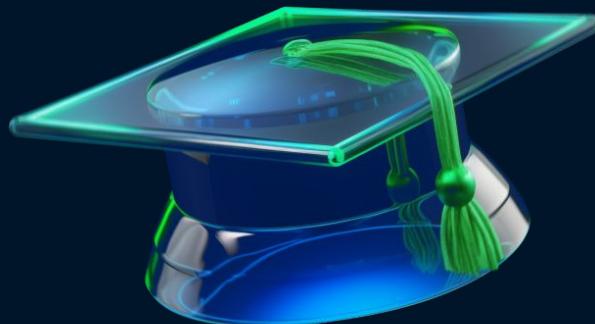
Quality Criteria for HTML Course

- ❤️ Mandatory for passing the course
- 💛 Required for the highest grade
- 💚 Optional

- 1.19. Each form element should have a <label>.
- 1.20. Resizing <textarea> should not break the layout.
- 1.21. Basic HTML form validation should be implemented.

Summary

1. The importance of forms on websites.
2. Components of forms.
3. How to solve styling issues with form elements.
4. How to validate form data.
5. Interactive tags outside of forms.

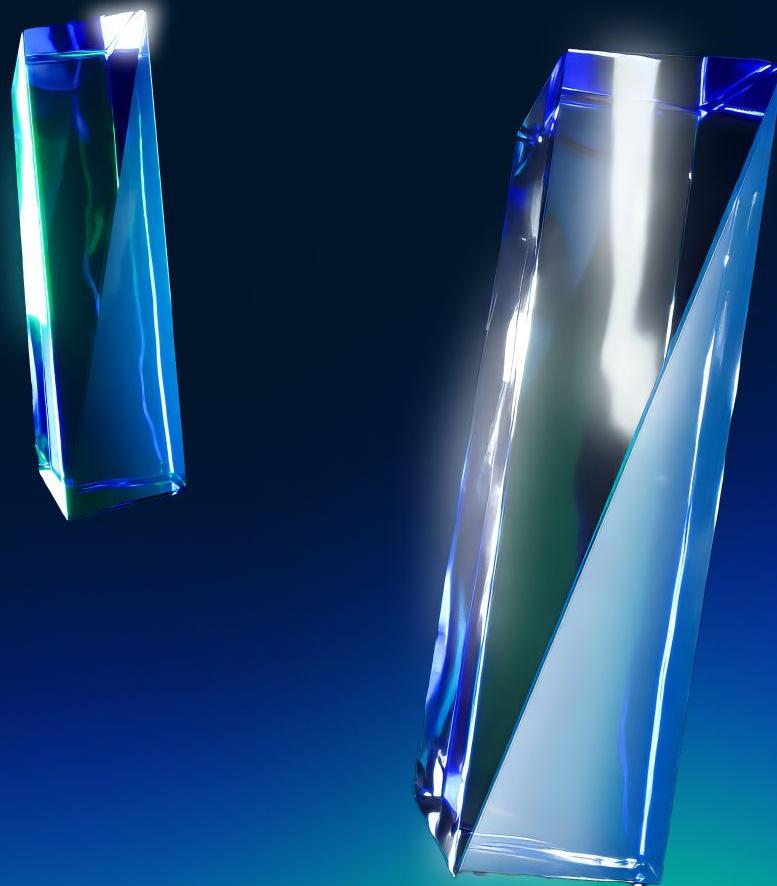


Homework

1. Complete the Lesson about forms elements
2. Markup and Style Forms in Binabox:
 - Specify Button Types
 - Validate Your Forms



QUESTIONS?



**Please fill out the feedback form
It's very important for us**



THANK YOU!
Have a good evening!