HTML Forms

WEB DEVELOPMENT · LESSON

Building a Sign-Up Page, One Line at a Time

A complete beginner's walkthrough of every tag and every attribute in a working sign-up form.

```
<form>
  <label>
  <input>
  <input>
  <submit>
</form>
```

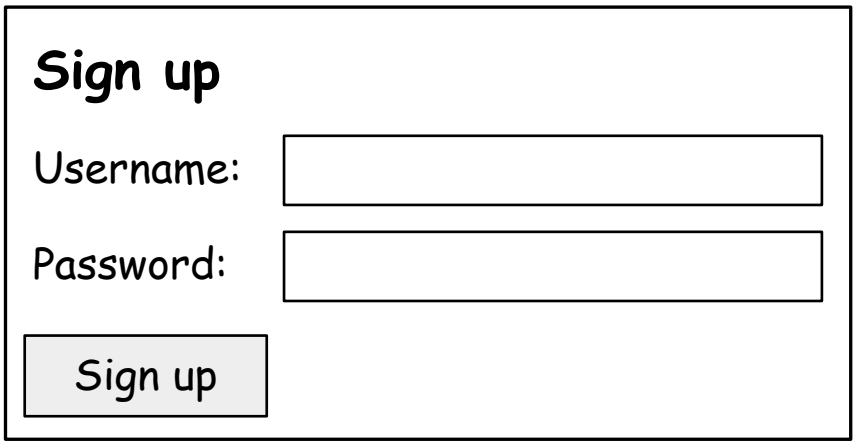
BY THE END OF THIS LESSON

Learning objectives

1	What an HTML form is, and why every website needs one
2	The three building blocks: <form>, <label> and <input>
3	What every single attribute in the code does
4	Answer exam-style questions on form, writing and interpreting code by hand

OUR PROJECT

Signup.html This is what we are building



A sign-up form as the visitor sees it in the browser.

A heading	The words 'Sign Up' at the top.
Two labels	The words 'Username:' and 'Password:'.
Two boxes	The empty rectangles where the visitor types.
One button	The small 'Sign up' button that sends everything.

OUR PROJECT

The full HTML – signup.html

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5    <meta name="viewport" content="width=device-width, initial-scale=1.0">
6    <title>Sign Up</title>
7  </head>
8  <body>
9    <h1>Sign Up</h1>
10   <form action="signup.php" method="post">
11     <label for="user">Username:</label>
12     <input type="text" id="user" name="user">
13    <br><br>
14     <label for="pw">Password:</label>
15     <input type="password" id="pw" name="pw">
16    <br><br>
17     <input type="submit" value="Sign up">
18   </form>
19 </body>
20 </html>
```

LINE 1 · ATTRIBUTE 1 OF 2

action="signup.php" — where the data is sent

action is the destination address of the filled-in form.

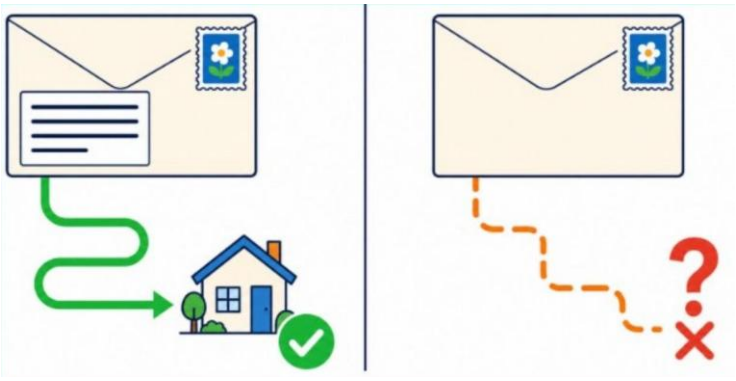
```
1 <form action="signup.php" method="post">
2   <label for="user">Username:</label>
3   <input type="text" id="user" name="user">
4   <label for="pw">Password:</label>
5   <input type="password" id="pw" name="pw">
6   <input type="submit" value="Sign up">
7 </form>
```

LINE 1 OF 7 · FIRST ATTRIBUTE

When the visitor clicks the button, the browser needs to know where to deliver what was typed. The action attribute answers that question by naming a file: signup.php.

That file sits on the web server. Inside it is code that receives the username and password, checks them, and saves the new account into a database.

Change this file name and the data goes somewhere else.



THINK OF IT LIKE THIS

The address written on the front of an envelope. A letter with no address never arrives anywhere.

LINE 1 · ATTRIBUTE 2 OF 2

method="post" — how the data travels

---

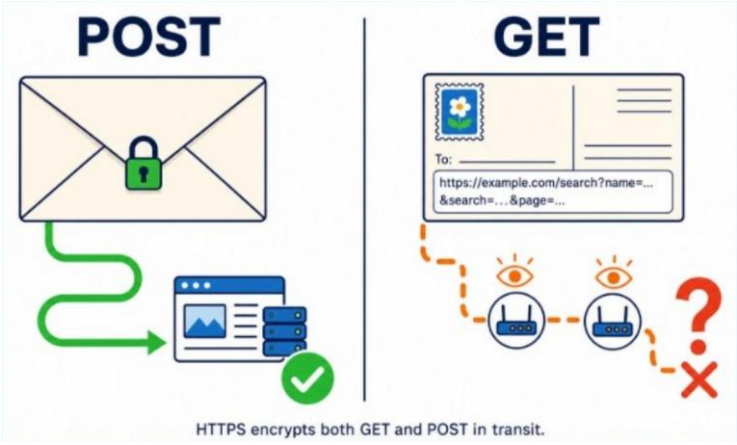
method chooses the way the typed values are carried to the server.

```
1 <form action="signup.php" method="post">
2   <label for="user">Username:</label>
3   <input type="text" id="user" name="user">
4   <label for="pw">Password:</label>
5   <input type="password" id="pw" name="pw">
6   <input type="submit" value="Sign up">
7 </form>
```

LINE 1 OF 7 · SECOND ATTRIBUTE

There are two choices. post tucks the values quietly inside the body of the request, where they are not displayed on screen. get instead pastes them onto the end of the web address in the address bar. Because this form carries a password, post is the correct and safe choice here.

Passwords must always travel by post, never by get.



THINK OF IT LIKE THIS

post is a sealed envelope. get is a postcard — anyone who handles it can read it on the way.

LINE 1 · GOING DEEPER

post and get, side by side

method="post"	**method="get"**
Use this for sign-up, log in, payments	Use this for searches and filters
• Values are hidden inside the request • The address bar stays clean • Can carry large amounts of data • Safe for passwords and private details • Cannot be bookmarked or shared	• Values are pasted into the web address • Everything is visible in the address bar • Limited to a small amount of data • Never use it for passwords • Can be bookmarked and shared

Rule of thumb: if the data is private or changes something on the server, use post.

The visible words sit between the two tags.



THINK OF IT LIKE THIS

post is a sealed envelope. get is a postcard — anyone who handles it can read it on the way.

LINE 2 · GOING DEEPER

Three reasons labels are not optional

1 Clarity	2 A bigger click target	3 Screen readers
An empty box on its own means nothing. The label tells the visitor exactly what belongs in it.	Because for="user" links them, clicking the word Username: places the cursor into the box. That is a real help on a phone.	Visually impaired users hear the page read aloud. A linked label is read out with the box; an unlinked box is announced as an unnamed edit field.

Accessibility is not an extra feature.
It is part of writing correct HTML.

LINE 3 · THE FIRST INPUT

<input> — the box the visitor types into

One <input> tag draws one box on the screen.

```
1 <form action="signup.php" method="post">
2   <label for="user">Username:</label>
3    <input type="text" id="user" name="user">
4   <label for="pw">Password:</label>
5   <input type="password" id="pw" name="pw">
6   <input type="submit" value="Sign up">
7 </form>
```

LINE 3 OF 7

Notice something unusual: there is no </input> anywhere. <input> is a self-closing tag, because it has no content to wrap around. It simply is the box. This is a void element.

Three attributes are packed inside this one line, and each does a completely different job.

TASK 1

Quick Quiz

Ten questions on the first three lines of your sign-up form

Write A, B, C or D for each question.

THE LINES UNDER TEST

```
1 <form action="signup.php" method="post">
2   <label for="user">Username:</label>
3   <input type="text" id="user" name="user">
```

SET 1 OF 2 · QUESTIONS

Questions 1 to 5

1. Which tag creates the container that holds the whole form together?

A	<container>
B	<form>
C	<input>
D	<label>

Answer: _____

2. In line 1, what is the job of action="signup.php"?

A	It sets the colour of the button
B	It names the file that will receive the data
C	It decides what the visitor may type
D	It closes the form

Answer: _____

3. Why does this form use method="post" rather than method="get"?

A	post is faster than get
B	post keeps the typed values out of the address bar
C	post is the only method PHP understands
D	post makes the boxes wider

Answer: _____

4. In <label for="user">Username:</label>, which part does the visitor actually see on screen?

A	for="user"
B	Username:
C	The word label
D	Every character in the line

Answer: _____

5. The for="user" written on the label must match which attribute on the input?

A	name
B	type
C	id
D	value

Answer: _____

SET 2 OF 2 · QUESTIONS

Questions 6 to 10

6. How does the line <input type="text" id="user" name="user"> get closed?

A	With </input>
B	With </text>
C	It does not — <input> is self-closing
D	With a second <input>

Answer: _____

7. What does type="text" tell the browser to do?

A	Make the writing bold
B	Draw a plain single-line box with visible characters
C	Send the text to the server
D	Print the word 'text' inside the box

Answer: _____

8. Which attribute becomes the key the server uses to read what was typed?

A	id
B	name
C	type
D	for

Answer: _____

9. A student deletes name="user" from line 3. What happens?

A	The box disappears from the page
B	The browser shows an error message
C	The box still works, but its value is never sent to the server
D	The page refuses to load

Answer: _____

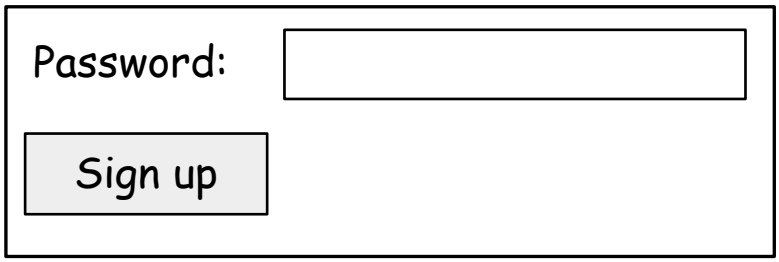
10. A student writes <label for="username"> but leaves id="user" on the input. What breaks?

A	Nothing at all
B	The input box disappears
C	The label and box are no longer linked, so clicking the label does nothing
D	The data is sent to the wrong file

Answer: _____

LINE 4 · THE SECOND LABEL

<label for="pw">Password:</label>



The pattern from line 2 simply repeats, with new words.

```
1 <form action="signup.php" method="post">
2   <label for="user">Username:</label>
3   <input type="text" id="user" name="user">
4    <label for="pw">Password:</label>
5   <input type="password" id="pw" name="pw">
6   <input type="submit" value="Sign up">
7 </form>
```

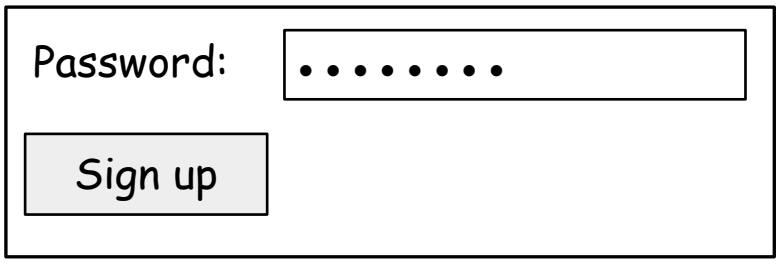
LINE 4 OF 7

Nothing new is being introduced here. The visible text is now Password:, and the pointer is now for="pw", which will match the id of the box on the next line.

Same shape as line 2 — only the words changed.

LINE 5 · THE PASSWORD BOX

type="password" — hiding what is typed



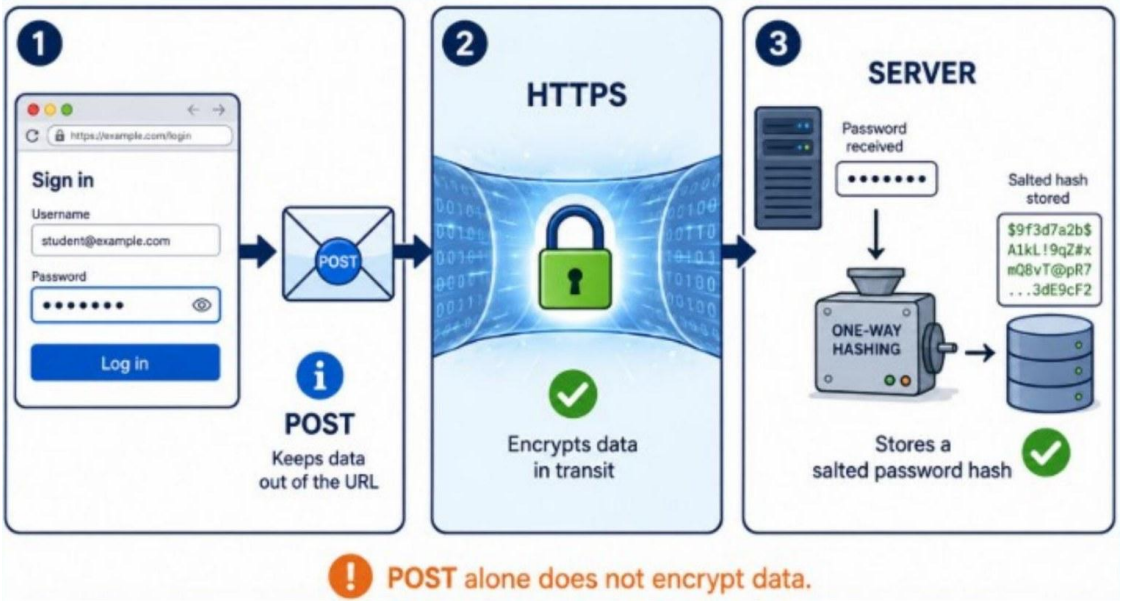
password draws the same box; it shows dots instead of characters.

```
1 <form action="signup.php" method="post">
2   <label for="user">Username:</label>
3   <input type="text" id="user" name="user">
4   <label for="pw">Password:</label>
5    <input type="password" id="pw" name="pw">
6   <input type="submit" value="Sign up">
7 </form>
```

LINE 5 OF 7 · FIRST ATTRIBUTE

The visitor types normally and the real value is stored and sent normally. Only the display is masked, so that somebody standing behind the visitor cannot read the password off the screen.

Masking is a screen shield, not encryption.



LINE 5 · THE REMAINING ATTRIBUTES

id="pw" and name="pw" — the same two jobs

Every box needs its own unique id and its own name.

```
1 <form action="signup.php" method="post">
2   <label for="user">Username:</label>
3   <input type="text" id="user" name="user">
4   <label for="pw">Password:</label>
5    <input type="password" id="pw" name="pw">
6   <input type="submit" value="Sign up">
7 </form>
```

LINE 5 OF 7 · SECOND AND THIRD ATTRIBUTES

LINE 6 · THE BUTTON TEXT

value="Sign up" — the words printed on the button

On a submit button, value is simply the caption the visitor reads.

```
1 <form action="signup.php" method="post">
2   <label for="user">Username:</label>
3   <input type="text" id="user" name="user">
4   <label for="pw">Password:</label>
5   <input type="password" id="pw" name="pw">
6    <input type="submit" value="Sign up">
7 </form>
```

LINE 6 OF 7 · SECOND ATTRIBUTE

Compare this with the screenshot: the small button says Sign up, and here in the code is exactly where those words come from.

One warning worth remembering. On a text box, value means something entirely different — there it fills the box with starting text before the visitor types anything.

Change it to value="Register" and the button reads Register.



LINE 6 · THE BUTTON TEXT

value="Sign up" — the words printed on the button

On a submit button, if there is no value, then the button just says Submit.




```
1 <form action="signup.php" method="post">
2   <label for="user">Username:</label>
3   <input type="text" id="user" name="user">
4   <label for="pw">Password:</label>
5   <input type="password" id="pw" name="pw">
6    <input type="submit" value="Sign up">
7 </form>
```

LINE 6 OF 7 · SECOND ATTRIBUTE

LINE 7 · THE CLOSING TAG

</form> — closing the container

Everything that must be sent has to sit above this line.

```
1 <form action="signup.php" method="post">
2   <label for="user">Username:</label>
3   <input type="text" id="user" name="user">
4   <label for="pw">Password:</label>
5   <input type="password" id="pw" name="pw">
6   <input type="submit" value="Sign up">
7 </form>
```

LINE 7 OF 7

The slash in </form> tells the browser that the container opened on line 1 ends here. Boxes written after this line still appear on the page, but they are outside the form and are silently ignored when the button is clicked.

Forgetting this single line is one of the most common beginner errors, and it produces a form that looks perfect and does nothing.

The forward slash is what makes a tag a closing tag.

PUTTING IT TOGETHER

What happens the moment you click Sign up

1 Collect	2 Package	3 Deliver	4 Respond
The browser looks inside <form> and finds every box that has a name attribute.	Each box becomes a pair: user=ada and pw=secret123.	The package is sent to signup.php using the post method.	The server reads the values, saves the account, and sends back a new page.

All four steps happen automatically. You did not write a single line of JavaScript.

GOING FURTHER

Change one word, get a different control

The code	What the browser draws	Typical use
type="text"	A plain single-line box	Names, cities, job titles
type="password"	Characters appear as dots	Passwords and PIN codes
type="email"	Checks that an @ is present	Email addresses
type="number"	Digits only, with up and down arrows	Age, quantity, price
type="date"	Opens a small calendar to pick from	Date of birth, booking dates
type="checkbox"	A tick box; several may be ticked	Interests, terms and conditions
type="radio"	Round buttons; only one may be chosen	Gender, delivery option
type="submit"	A clickable button that sends the form	Sign up, Log in, Search

Everything else in the line — id, name and the closing bracket — stays exactly the same.

TROUBLESHOOTING

Four mistakes almost every beginner makes

! Forgetting name	! Forgetting </form>
The box appears and accepts typing, but the server receives nothing at all. Always ask: does every input have a name?	The form looks perfect on screen and the button does nothing useful. Check that the closing tag is there.
! for and id do not match	**! Using get for a password**
The label stops being clickable and screen readers lose the connection. Check the spelling character by character.	The password appears in the address bar for anyone to read. Sign-up and log-in forms always use post.

GOING FURTHER

Three small upgrades to this form

placeholder	required	minlength
placeholder="e.g. ada99"	required	minlength="8"
Faint grey example text inside the empty box. It disappears the moment the visitor starts typing.	The browser refuses to submit the form while the box is empty, and shows a warning by itself.	Blocks passwords that are too short before the data ever leaves the page.

```
<input type="text" id="user" name="user" placeholder="e.g. ada99" required>
```

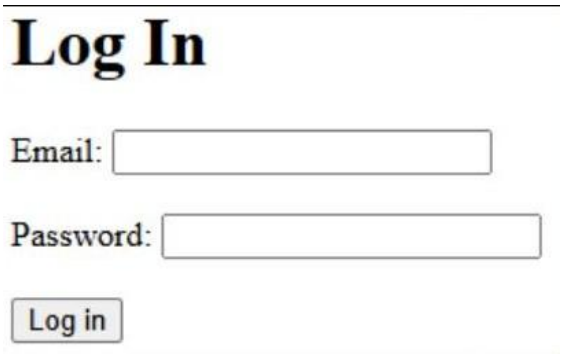
TASK 2

YOUR TURN

Practice task: build a log-in form

Working on your own, create a file called **login.html** containing a form that:

- sends its data to login.php using the post method
- asks for an email address using type="email"
- asks for a password that is hidden as dots
- links every label to its box with for and id
- has a button that reads Log in



Then open it in a browser and click the label text. Does the cursor jump into the box?

☐	**Marking checklist**	
☐	**Does every input have a name?**	Without it, nothing is sent.
☐	**Is there a closing </form>?**	The most common missing line.
☐	**Does each for match an id exactly?**	Check the spelling twice.
☐	**Is the method post, not get?**	Passwords must never show in the URL.

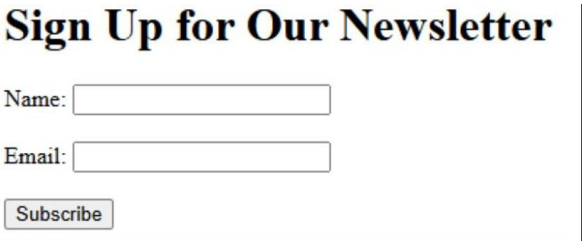
Write your code here:

TASK 3

Build a newsletter sign-up form

Working on your own, create a file called **newsletter.html** containing a form that:

- sends its data to subscribe.php using the post method
- asks for a name using type="text"
- asks for an email address using type="email"
- links every label to its box with for and id
- has a button that reads Subscribe



Then open it in a browser and click each label. Does the cursor jump into the box beside it?

Write your code here:

TASK 4

Build a give us your feedback form

Working on your own, create a file called **feedback.html** containing a form that:

- sends its data to feedback.php using the post method
- asks for an email address type= "email"
- asks for rating out of 10 using type="number"
- links every label to its box with for and id
- has a button that reads Send feedback



Then open it in a browser and click each label. Does the cursor jump into the box beside it?

Write your code here:

TASK 5

Build a book search form

Working on your own, create a file called **book.html** containing a form that:

- sends its data to books.php using the get method

- Have a box labelled Title: with name="title"
- Have a box labelled Author: with name="author"
- links every label to its box with for and id
- has a button that reads Find books

- **If a student types Things Fall Apart and Achebe, the address bar should read:**

  `books.php?title=Things+Fall+Apart&author=Achebe`

Then open it in a browser and click each label. Does the cursor jump into the box beside it?

Write your code here: