Forms & Inputs (text, radio, checkbox, select)

HTML forms explained: form and input tags, labels, text email and password fields, radio vs checkbox, select dropdowns, required and buttons.

Introduction

A page that only *shows* things is half a website. The moment a visitor must *give* something — sign up, log in, search, apply for admission — a form appears. Forms are the most-used HTML in real life: every account you have ever made began as one. In this lesson you build a complete signup form with every common field type, and you learn the small rules (labels, names, radio groups) that separate a form that works from a form that only looks fine. *(Roman Urdu: Signup ho ya admission — jahan user kuch deta hai, wahan form hota hai.)*

Key points

- A **form** collects information from the visitor — signup, login, search, admission. All fields live inside `<form>...</form>`.
- `<input>` is the main field tag. Its **type** decides what kind of field it is: `<input>` has **no closing tag**.
  - `type="text"` — a single line of text.
  - `type="email"` — an email address (the browser checks the format).
  - `type="password"` — hidden typing (dots, not letters).
  - `type="number"` — numbers only.
- The **name** attribute is the field's identity: when the form is sent, every answer travels with its field's name. A field without a `name` sends nothing. Think of `name` as the label written on an envelope — without it, the server receives answers but cannot tell which answer belongs to which question.
- `<label>` is the field's title text. Connect it with `for` and the input's `id`: `<label for="fullname">Full Name</label>` + `<input id="fullname" ...>` **Clicking the label selects the field** (the lesson's label-focus demo) — and screen readers read the label out loud.
- **Radio buttons** (`type="radio"`) = choose **ONE** from a group. All radios of one group must share the **same** `name` — the shared name is what makes them one group, so only one can be selected.
- **Checkboxes** (`type="checkbox"`) = choose **MANY**. Each checkbox is its own choice, like subjects or hobbies.

  (Roman Urdu: Radio = sirf aik chunein, checkbox = jitne chaaho chunein.)
- `<select>` makes a **dropdown** for a long list of choices (like cities). Each choice is an `<option>`.
- `required` on a field: the browser **stops the submit and warns** if the field is empty. Free checking, no extra code.
- **Buttons:** `<button type="submit">` sends the form, `type="reset"` clears all fields, `type="button"` is a plain button that does nothing by itself. Careful: a button inside a form is a **submit** button by default. So

choose the type on purpose, every time — an untyped button quietly submits the form when all you wanted was a plain click.

Code examples

Example 1 — A small signup form (the lesson's build, in short)

```
<form>
  <label for="fullname">Full Name</label>
  <input type="text" id="fullname" name="fullname" required>

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

  <p>Gender:</p>
  <label><input type="radio" name="gender" value="male"> Male</label>
  <label><input type="radio" name="gender" value="female"> Female</label>

  <p>Subjects:</p>
  <label><input type="checkbox" name="subject" value="math"> Math</label>
  <label><input type="checkbox" name="subject" value="computer"> Computer</label>

  <label for="city">City</label>
  <select id="city" name="city">
    <option>Lahore</option>
    <option>Karachi</option>
    <option>Multan</option>
  </select>

  <button type="submit">Sign Up</button>
</form>
```

What it does: A complete signup form — name and email (both required), one gender (radio group with one shared name), many subjects (checkboxes), a city dropdown, and a submit button. Click any label and its field gets selected.

What you will see: labelled boxes in a column; two round gender buttons where selecting one un-selects the other; two square subject boxes that can both stay ticked; a City box that opens a small list when clicked. Press "Sign Up" with the name empty and the browser refuses, pointing at the empty field — that is `required` doing its job.

Example 2 — A login form (email + password)

```
<form>
  <label for="login-email">Email</label>
  <input type="email" id="login-email" name="email" required>

  <label for="login-pass">Password</label>
  <input type="password" id="login-pass" name="password" required>

  <button type="submit">Log In</button>
  <button type="reset">Clear</button>
</form>
```

What it does: The two-field form behind every login page on the internet. The email field checks the address format, the password field hides the typing, and the form offers two different buttons on purpose: submit sends, reset only clears.

What you will see: as you type the password, the box shows dots instead of letters — nobody standing behind you can read it. Type something in both boxes and press "Clear": both boxes empty themselves, and nothing is sent anywhere.

Example 3 — A dropdown with a must-tick checkbox

```
<form>
  <label for="course">Choose your course</label>
  <select id="course" name="course">
    <option>Web Development</option>
    <option>Graphic Design</option>
    <option>Digital Marketing</option>
  </select>

  <label><input type="checkbox" name="agree" required> I agree to the class rules</label>

  <button type="submit">Join</button>
</form>
```

What it does: A tiny "join a course" form. The `<select>` keeps a long choice list inside one small box, and the checkbox carries `required` — the form cannot be sent until the visitor ticks it.

What you will see: one compact dropdown box; click it and the three courses open as a list. Press "Join" without ticking the agreement box and the browser stops you with a warning on the checkbox — `required` works on checkboxes too, not only on text fields.

Quick recap

- `<form>` collects the visitor's answers; `<input>` is the main field, and its `type` decides the field's kind (text, email, password, number…).
- No `name` = the answer is never sent. `name` is the field's identity.
- `<label for="...">` + the input's `id` connect label and field — clicking the label selects the field.
- **Radio = one choice** (whole group shares one `name`); **checkbox = many choices**.
- `<select>` + `<option>` = a dropdown for long lists; `required` blocks an empty submit for free.
- Buttons: `submit` sends, `reset` clears, `button` does nothing alone — and a button in a form submits by default.

Common mistakes

1. **Radios with different names** — `name="male"` and `name="female"` make two separate groups, so both can be selected at once. A radio group = **one shared name**.
2. **No labels** — the visitor guesses what each box wants, and screen-reader users hear nothing. Every field needs a `<label>`.
3. **Password written as `type="text"`** — the password shows as plain letters on the screen. Password fields must be `type="password"`.
4. **Missing `name` attributes** — the form looks fine, but on submit those answers are never sent.

Practice

1. Build an **admission form**: full name, email, password, gender (radio), favourite subjects (checkboxes), city (select). Put `required` on name and email, and test the warning by submitting with an empty name.
2. Click every label in your form. Each click should select its field. If one label does nothing, check its `for` and the input's `id` — they must match exactly.

Practice quiz

1. What does <form> do?

- Shows a picture
- Collects input from the user
- Makes a heading
- Plays sound

Answer: Collects input from the user

2. What does <button> do?

- A heading

- A clickable button
- A list
- An image

Answer: A clickable button

Watch the video lesson: https://youtu.be/g4AggNguulQ

Practice tool: CV Builder — https://scopetreasure.com/tool/cv-builder

Read online: https://scopetreasure.com/notes/html-forms-and-inputs · More free tools and notes at https://scopetreasure.com