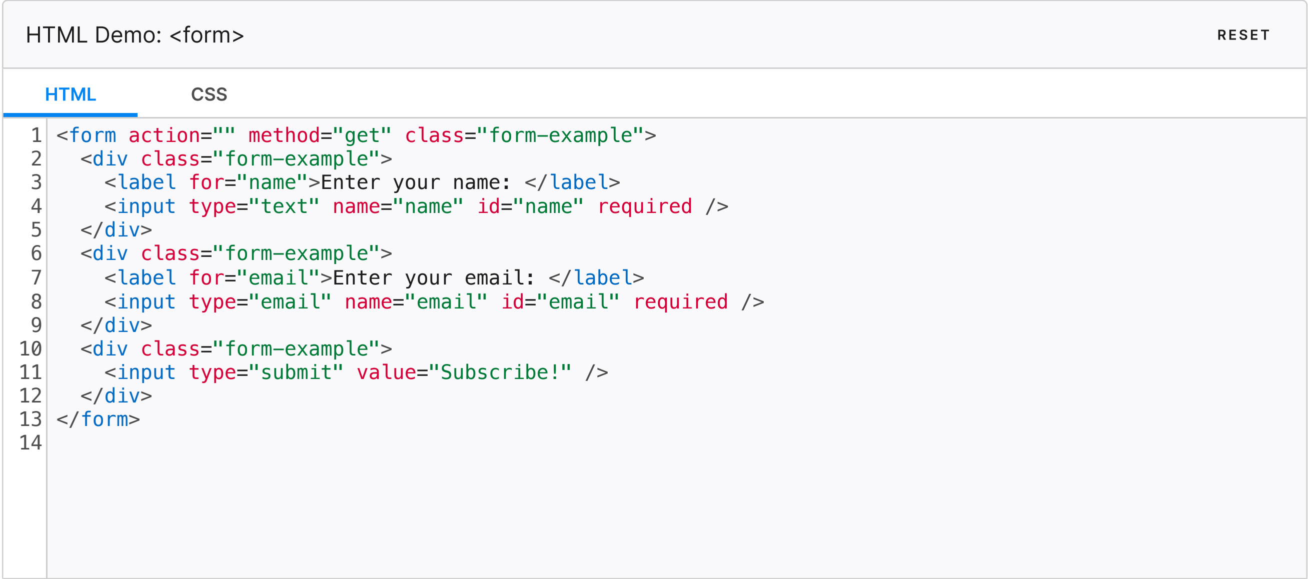
mdn web docs

<form>: The Form element

Baseline Widely available

The `<form>` HTML element represents a document section containing interactive controls for submitting information.

Try it



HTML Demo: <form>

RESET

HTML CSS

```html
<form action="" method="get" class="form-example">
  <div class="form-example">
    <label for="name">Enter your name: </label>
    <input type="text" name="name" id="name" required />
  </div>
  <div class="form-example">
    <label for="email">Enter your email: </label>
    <input type="email" name="email" id="email" required />
  </div>
  <div class="form-example">
    <input type="submit" value="Subscribe!" />
  </div>
</form>
```

It is possible to use the `:valid` and `:invalid` CSS pseudo-classes to style a `<form>` element based on whether the `elements` inside the form are valid.

Attributes

This element includes the global attributes.

`accept` 🗑

Comma-separated content types the server accepts.

> **Note: This attribute has been deprecated and should not be used.** Instead, use the `accept` attribute on `<input type=file>` elements.

`accept-charset`

The character encoding accepted by the server. The specification allows a single case-insensitive value of `"UTF-8"`, reflecting the ubiquity of this encoding (historically multiple character encodings could be specified as a comma-separated or space-separated list).

`autocapitalize`

Controls whether inputted text is automatically capitalized and, if so, in what manner. See the `autocapitalize` global attribute page for more information.

`autocomplete`

Indicates whether input elements can by default have their values automatically completed by the browser. `autocomplete` attributes on form elements override it on `<form>`. Possible values:

- `off`: The browser may not automatically complete entries. (Browsers tend to ignore this for suspected login forms; see Managing autofill for login fields.)
- `on`: The browser may automatically complete entries.

`name`

The name of the form. The value must not be the empty string, and must be unique among the `form` elements in the forms collection that it is in, if any.

`rel`

Controls the annotations and what kinds of links the form creates. Annotations include `external`, `nofollow`, `opener`, `noopener`, and `noreferrer`. Link types include `help`, `prev`, `next`, `search`, and `license`. The `rel` value is a space-separated list of these enumerated values.

Attributes for form submission

The following attributes control behavior during form submission.

`action`

The URL that processes the form submission. This value can be overridden by a `formaction` attribute on a `<button>`, `<input type="submit">`, or `<input type="image">` element. This attribute is ignored when `method="dialog"` is set.

`enctype`

If the value of the `method` attribute is `post`, `enctype` is the MIME type of the form submission. Possible values:

- `application/x-www-form-urlencoded`: The default value.
- `multipart/form-data`: Use this if the form contains `<input>` elements with `type=file`.
- `text/plain`: Useful for debugging purposes.

This value can be overridden by `formenctype` attributes on `<button>`, `<input type="submit">`, or `<input type="image">` elements.

`method`

The HTTP method to submit the form with. The only allowed methods/values are (case insensitive):

- `post`: The `POST` method; form data sent as the request body.
- `get` (default): The `GET`; form data appended to the `action` URL with a `?` separator. Use this method when the form has no side effects.
- `dialog`: When the form is inside a `<dialog>`, closes the dialog and causes a `submit` event to be fired on submission, without submitting data or clearing the form.

This value is overridden by `formmethod` attributes on `<button>`, `<input type="submit">`, or `<input type="image">` elements.

`novalidate`

This Boolean attribute indicates that the form shouldn't be validated when submitted. If this attribute is not set (and therefore the form ***is*** validated), it can be overridden by a `formnovalidate` attribute on a `<button>`, `<input`

`type="submit">`, or `<input type="image">` element belonging to the form.

`target`

Indicates where to display the response after submitting the form. It is a name/keyword for a *browsing context* (for example, tab, window, or iframe). The following keywords have special meanings:

- `_self` (default): Load into the same browsing context as the current one.
- `_blank`: Load into a new unnamed browsing context. This provides the same behavior as setting `rel="noopener"` which does not set `window.opener`.
- `_parent`: Load into the parent browsing context of the current one. If no parent, behaves the same as `_self`.
- `_top`: Load into the top-level browsing context (i.e., the browsing context that is an ancestor of the current one and has no parent). If no parent, behaves the same as `_self`.
- `_unfencedTop`: Load the response from a form inside an embedded fenced frame into the top-level frame (i.e., traversing beyond the root of the fenced frame, unlike other reserved destinations). Only available inside fenced frames.

This value can be overridden by a `formtarget` attribute on a `<button>`, `<input type="submit">`, or `<input type="image">` element.

Examples

HTML Play

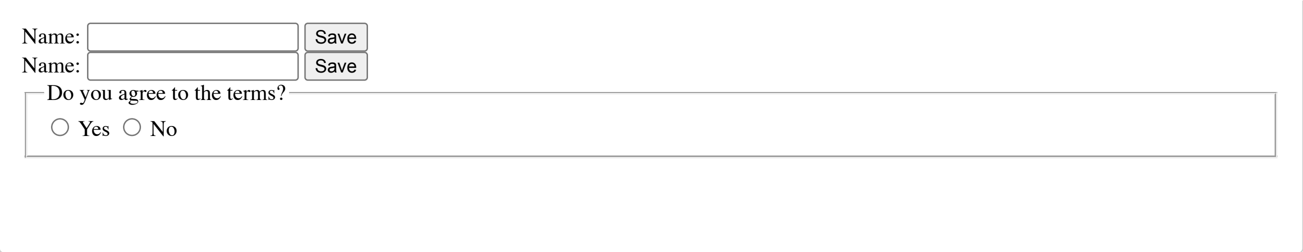
```html
<!-- Form which will send a GET request to the current URL -->
<form method="get">
  <label>
    Name:
    <input name="submitted-name" autocomplete="name" />
  </label>
  <button>Save</button>
</form>

<!-- Form which will send a POST request to the current URL -->
<form method="post">
  <label>
    Name:
    <input name="submitted-name" autocomplete="name" />
  </label>
  <button>Save</button>
</form>

<!-- Form with fieldset, legend, and label -->
<form method="post">
  <fieldset>
    <legend>Do you agree to the terms?</legend>
    <label><input type="radio" name="radio" value="yes" /> Yes</label>
    <label><input type="radio" name="radio" value="no" /> No</label>
  </fieldset>
</form>
```

Result

Play



Name: [ ] Save
Name: [ ] Save

Do you agree to the terms?
○ Yes ○ No

Technical summary

Content categories	Flow content, palpable content
Permitted content	Flow content, but not containing `<form>` elements
Tag omission	None, both the starting and ending tag are mandatory.
Permitted parents	Any element that accepts flow content
Implicit ARIA role	`form`
Permitted ARIA roles	`search`, `none` or `presentation`
DOM interface	`HTMLFormElement`

Specifications

Specification
HTML Standard # the-form-element

Browser compatibility

Report problems with this compatibility data on GitHub

	Chrome	Edge	Firefox	Opera	Safari	Chrome Android	Firefox for Android	Opera Android	Safari on iOS	Samsung Internet	WebView Android	WebView on iOS
`form`	✓ 1	✓ 12	✓ 1	✓ 15	✓ 3	✓ 18	✓ 4	✓ 14	✓ 2	✓ 1.0	✓ 4.4	✓ 2
`accept-charset`	✓ 1	✓ 12	✓ 1	✓ 15	✓ 3	✓ 18	✓ 4	✓ 14	✓ 2	✓ 1.0	✓ 4.4	✓ 2
`action`	✓ 1	✓ 12	✓ 1	✓ 15	✓ 3	✓ 18	✓ 4	✓ 14	✓ 2	✓ 1.0	✓ 4.4	✓ 2
`autocomplete`	✓ 14 ✱	✓ 12	✓ 4	✓ 15 ✱	✓ 6	✓ 18 ✱	✓ 4	✓ 14 ✱	✓ 6	✓ 1.0 ✱	✓ 4.4 ✱	✓ 6
`new-password` value	☒ No	☒ No	✓ 67	☒ No	☒ No	☒ No	✓ 67	☒ No	☒ No	☒ No	☒ No	☒ No
`one-time-code` value	✓ 93	✓ 93	✓ 109	✓ 79	✓ 12	✓ 84	✓ 109	✓ 60	✓ 12	✓ 14.0	☒ No	✓ 12

	🖥					📱						
	Chrome	Edge	Firefox	Opera	Safari	Chrome Android	Firefox for Android	Opera Android	Safari on iOS	Samsung Internet	WebView Android	WebView on iOS
`webauthn` ⚗ value	✓ 108	✓ 108	⊗ No	✓ 94	⊗ No	✓ 108	⊗ No	✓ 73	⊗ No	✓ 21.0	✓ 108	⊗ No
`enctype`	✓ 1	✓ 12	✓ 1	✓ 15	✓ 3	✓ 18	✓ 4	✓ 14	✓ 2	✓ 1.0	✓ 4.4	✓ 2
`method`	✓ 1	✓ 12	✓ 1	✓ 15	✓ 3	✓ 18	✓ 4	✓ 14	✓ 2	✓ 1.0	✓ 4.4	✓ 2
`name`	✓ 1	✓ 12	✓ 1	✓ 15	✓ 3	✓ 18	✓ 4	✓ 14	✓ 2	✓ 1.0	✓ 4.4	✓ 2
`novalidate`	✓ 10	✓ 12	✓ 4	✓ 15	✓ 10.1	✓ 18	✓ 4	✓ 14	✓ 10.3	✓ 1.0	✓ 37	✓ 10.3
`rel`	✓ 108	✓ 108	✓ 111	✓ 94	✓ 15.4	✓ 108	✓ 111	✓ 73	✓ 15.4	✓ 21.0	✓ 108	✓ 15.4
`target`	✓ 1	✓ 12	✓ 1	✓ 15	✓ 3	✓ 18	✓ 4	✓ 14	✓ 2	✓ 1.0	✓ 4.4	✓ 2

Tip: you can click/tap on a cell for more information.

✓ Full support ⊗ No support ⚗ Experimental. Expect behavior to change in the future. ✱ See implementation notes.

See also

- HTML forms guide
- Other elements that are used when creating forms: `<button>`, `<datalist>`, `<fieldset>`, `<input>`, `<label>`, `<legend>`, `<meter>`, `<optgroup>`, `<option>`, `<output>`, `<progress>`, `<select>`, `<textarea>`.
- Getting a list of the elements in the form: `HTMLFormElement.elements`
- ARIA: Form role
- ARIA: Search role

Help improve MDN

Was this page helpful to you?

Yes No

Learn how to contribute.

This page was last modified on Nov 20, 2024 by MDN contributors.