<dialog>: The Dialog element

The `<dialog>` HTML element represents a dialog box or other interactive component, such as a dismissible alert, inspector, or subwindow.

Attributes

This element includes the global attributes.

> **Warning:** The `tabindex` attribute must not be used on the `<dialog>` element.

`open`

Indicates that the dialog is active and can be interacted with. When the `open` attribute is not set, the dialog *shouldn't* be shown to the user. It is recommended to use the `.show()` or `.showModal()` methods to render dialogs, rather than the `open` attribute. If a `<dialog>` is opened using the `open` attribute, it will be non-modal.

Accessibility considerations

The native HTML `<dialog>` element should be used in creating modal dialogs as it provides usability and accessibility features that must be replicated if using other elements for a similar purpose. Use the appropriate `.showModal()` or `.show()` method to render dialogs. If creating a custom dialog implementation, ensure all expected default behaviors are supported and proper labeling recommendations are followed.

When implementing a dialog, it is important to consider the most appropriate place to set user focus. Explicitly indicating the initial focus placement by use of the autofocus attribute will help ensure initial focus is set to the element deemed the best initial focus placement for any particular dialog. When in doubt, as it may not always be known where initial focus could be set within a dialog, particularly for instances where a dialog's content is dynamically rendered when invoked, then if necessary authors may

decide focusing the `<dialog>` element itself would provide the best initial focus placement. When using `HTMLDialogElement.showModal()` to open a `<dialog>`, focus is set on the first nested focusable element.

Ensure a mechanism is provided to allow users to close a dialog. The most robust way to ensure all users can close a dialog is to include an explicit button to do so. For instance, a confirmation, cancel or close button as appropriate. Additionally, for those using a device with a keyboard, the `Escape` key is commonly expected to close modal dialogs as well. By default, a `<dialog>` invoked by the `showModal()` method will allow for its dismissal by the `Escape`. A non-modal dialog does not dismiss via the `Escape` key by default, and depending on what the non-modal dialog represents, it may not be desired for this behavior. If multiple modal dialogs are open, `Escape` should only close the last shown dialog. When using `<dialog>`, this behavior is provided by the browser.

The `<dialog>` element is exposed by browsers similarly to custom dialogs using the ARIA role="dialog" attribute. `<dialog>` elements invoked by the `showModal()` method will have an implicit aria-modal="true", whereas `<dialog>` elements invoked by the `show()` method, or rendered by use of the `open` attribute or changing the default `display` of a `<dialog>` will be exposed as `[aria-modal="false"]`. When implementing modal dialogs, everything other than the `<dialog>` and its contents should be rendered inert using the `inert` attribute. When using `<dialog>` along with the `HTMLDialogElement.showModal()` method, this behavior is provided by the browser.

Usage notes

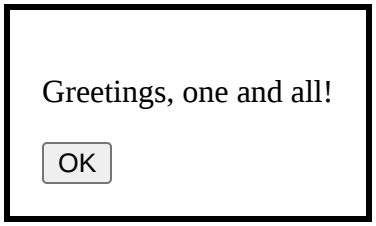
- `<form>` elements can close a `<dialog>` if they have the attribute `method="dialog"` or if the button used to submit the form has `formmethod="dialog"` set. In this case, the state of the form controls are saved, not submitted, the `<dialog>` closes, and the `returnValue` property gets set to the `value` of the button that was used to save the form's state.
- The `::backdrop` CSS pseudo-element can be used to style the backdrop that is displayed behind a `<dialog>` element when the dialog is displayed with `HTMLDialogElement.showModal()`. For example, to dim unreachable content behind the modal dialog.

Examples

Simple example

The following will render a non-modal, or modal-less, dialog. The "OK" button allows the dialog to be closed when activated.

```
<dialog open>
  <p>Greetings, one and all!</p>
  <form method="dialog">
    <button>OK</button>
  </form>
</dialog>
```



Because this dialog was opened via the `open` attribute, it is non-modal. In this example, when the dialog is dismissed, no method is provided to re-open it. Opening dialogs via `HTMLDialogElement.show()` is preferred over the toggling of the boolean `open` attribute.

Advanced example

This example opens a modal dialog when the "Show the dialog" button is activated. The dialog contains a form. Updating the value of the `<select>` updates the value of the "confirm" button.

HTML

```
<!-- A modal dialog containing a form -->
<dialog id="favDialog">
  <form method="dialog">
    <p>
      <label>Favorite animal:
        <select>
          <option value="default">Choose…</option>
          <option>Brine shrimp</option>
          <option>Red panda</option>
          <option>Spider monkey</option>
        </select>
      </label>
    </p>
```

```
    <div>
      <button value="cancel">Cancel</button>
      <button id="confirmBtn" value="default">Confirm</button>
    </div>
  </form>
</dialog>
<p>
  <button id="showDialog">Show the dialog</button>
</p>
<output></output>
```

JavaScript

```
const showButton = document.getElementById('showDialog');
const favDialog = document.getElementById('favDialog');
const outputBox = document.querySelector('output');
const selectEl = favDialog.querySelector('select');
const confirmBtn = favDialog.querySelector('#confirmBtn');

// "Update details" button opens the <dialog> modally
showButton.addEventListener('click', () => {
    favDialog.showModal();
});
// "Favorite animal" input sets the value of the submit button
selectEl.addEventListener('change', (e) => {
  confirmBtn.value = selectEl.value;
});
// "Confirm" button of form triggers "close" on dialog because of [method="dialog"]
favDialog.addEventListener('close', () => {
  outputBox.value = `ReturnValue: ${favDialog.returnValue}.`;
});
```

Result

Show the dialog

Note the JavaScript did not include the `HTMLDialogElement.close()` method. When a form in a dialog is submitted, if the method is `dialog`, the current state of the form is saved, not submitted, and the dialog gets closed.

It is important to provide a mechanism to close a dialog within the `dialog` element. For instance, the `Esc` key does not close non-modal dialogs by default, nor can one assume that a user will even have access to a physical keyboard (e.g., someone using a touch screen device without access to a keyboard).

Technical summary

Content categories	Flow content, sectioning root
Permitted content	Flow content
Tag omission	None, both the starting and ending tag are mandatory.
Permitted parents	Any element that accepts flow content
Implicit ARIA role	dialog
Permitted ARIA roles	`alertdialog`
DOM interface	`HTMLDialogElement`

Specifications

Specification
HTML Standard # the-dialog-element

Browser compatibility

See also

- The `close` event
- The `cancel` event
- HTML forms guide.
- The `::backdrop` pseudo-element
- dialog-polyfill

This page was last modified on Jan 19, 2023 by MDN contributors.