JavaScript Project - Modal Popup

Back to JS page

Table of Contents

- JavaScript Project - Modal Popup
  - Modal Popup Overview
  - What You Will Learn
  - First: Create the HTML
    - Example 1
  - Then: Create the CSS
    - Example 2
  - CSS Explained
  - Then: Create the JavaScript
    - Example 3
  - JavaScript Explained
  - Connect Functions to Buttons
  - Connect Function to Clicking Outside
  - Connect Function to Escape Key
  - Common Mistakes
  - Bonus Challenges (Level Up)
    - Example 4
  - Document
  - Reference

Modal Popup Overview

A **modal is a popup window** that appears on top of the main webpage overlaying the current view.

In this project, we will build a modal that can be closed in three ways:

1. Clicking the close button (`x`)
2. Clicking outside the modal container
3. Pressing the Escape key

What You Will Learn

- How to dynamically show and hide HTML elements
- How to add and remove CSS classes using JavaScript

- How to listen to click and keydown event targets
- How to detect clicks outside a bounding element box

First: Create the HTML

The modal layout consists of two main parts:

- **The overlay background:** Dims the page context behind.
- **The modal content box:** Displays the pop-up text and controls.

```
<!DOCTYPE html>
<html>
<body>

<h2>Modal Popup</h2>
<p>From w3schools.com, Experiment by Teeratus_R</p>

<button id="openBtn">Open Modal</button>

<div id="modal" class="modal-overlay">
  <div class="modal-box">
    <button id="closeBtn" class="modal-close">&times;</button>
    <h3>Hello!</h3>
    <p>This is a modal popup.</p>
  </div>
</div>

</body>
</html>
```

Example 1

Result View Example

Then: Create the CSS

Define structural position attributes to overlay the window and control visual overlays:

```
<style>
/* The overlay (background) */
.modal-overlay {
  display: none;
  position: fixed;
  left: 0;
  top: 0;
  width: 100%;
  height: 100%;
  background: rgba(0, 0, 0, 0.5);
}

/* Show the modal */
.modal-overlay.show {
  display: block;
}

/* The modal box */
.modal-box {
  background: white;
  width: 90%;
  max-width: 400px;
  margin: 100px auto;
  padding: 20px;
  border-radius: 10px;
  position: relative;
}

/* Close button */
.modal-close {
  position: absolute;
  right: 12px;
  top: 8px;
  font-size: 24px;
  border: none;
  background: none;
  cursor: pointer;
}
</style>
```

Example 2

Result View Example

CSS Explained

- **Hide modal by default:** We use `display: none` on `.modal-overlay`.
- **Toggle visibility:** We define a `.modal-overlay.show` selector specifying `display: block`. JavaScript will add and remove this helper class dynamically.

Then: Create the JavaScript

Select elements and implement toggle helpers mapping handlers to events:

```
<script>
const modal = document.getElementById("modal");
const openBtn = document.getElementById("openBtn");
const closeBtn = document.getElementById("closeBtn");

function openModal() {
  modal.classList.add("show");
}

function closeModal() {
  modal.classList.remove("show");
}

openBtn.addEventListener("click", openModal);
closeBtn.addEventListener("click", closeModal);

/* Close when clicking outside the modal box */
modal.addEventListener("click", function (event) {
  if (event.target === modal) {
    closeModal();
  }
});

/* Close when pressing Escape */
document.addEventListener("keydown", function (event) {
  if (event.key === "Escape") {
    closeModal();
  }
});
</script>
```

Example 3

Result View Example

JavaScript Explained

We use `classList.add("show")` and `classList.remove("show")` to adjust display states on triggers.

Connect Functions to Buttons

Register listeners to handle standard interaction click paths:

```
openBtn.addEventListener("click", openModal);
closeBtn.addEventListener("click", closeModal);
```

Connect Function to Clicking Outside

Check whether the target element triggering click events is the overlay container node itself:

```
modal.addEventListener("click", function (event) {
  if (event.target === modal) {
    closeModal();
  }
});
```

> [!NOTE]
> `event.target === modal` checks if click bubbles up/originates on the overlay background surface (outside the white `.modal-box`).

Connect Function to Escape Key

Listen globally for Escape keystrokes to allow quick exits:

```
document.addEventListener("keydown", function (event) {
  if (event.key === "Escape") {
    closeModal();
  }
});
```

Common Mistakes

- **Modal overlay does not close when clicked:** Check that you compare `event.target === modal` strictly.

- **Escape closes invisible modals:** You can optionally check `modal.classList.contains("show")` before hiding.
- **Display none overrides animation:** Animations require opacity transitions or fade keyframe steps rather than display swaps.

Bonus Challenges (Level Up)

Elevate the design with premium UX transitions and accessibility support:

- Add a fade-in and scale-up effect using CSS opacity and transform transition metrics.
- Move document focus inside the modal overlay dialog upon open.
- Lock/Trap tab key navigation focus cycles within active boundaries.
- Restore focus tracking state to previous active elements on exit.

```
<!-- Bonus Challenge Implementation -->
<script>
const modal = document.getElementById("modal");
const modalBox = modal.querySelector(".modal-box");
const openBtn = document.getElementById("openBtn");
const closeBtn = document.getElementById("closeBtn");

let previousActiveElement = null;

function openModal() {
  previousActiveElement = document.activeElement;
  modal.classList.add("show");
  modalBox.focus();
}

function closeModal() {
  modal.classList.remove("show");
  if (previousActiveElement) {
    previousActiveElement.focus();
  }
}

// Focus Trap inside the modal
modalBox.addEventListener("keydown", function (event) {
  const focusable = modalBox.querySelectorAll('button, [href], input, [tabindex]:not([tabindex="
  const first = focusable[0];
  const last = focusable[focusable.length - 1];

  if (event.key === "Tab") {
    if (event.shiftKey) {
      if (document.activeElement === first) {
        last.focus();
        event.preventDefault();
      }
    } else {
      if (document.activeElement === last) {
        first.focus();
        event.preventDefault();
      }
    }
  }
});
</script>
```

Example 4

Result View Example

Document

Document in project

You can Download PDF file.

Reference

- W3Schools JavaScript Modal Popup Project