Lesson 5: Keyboard & Mouse Events

🧠 Goal

Learn how to make your web pages react to **keyboard** and **mouse** actions. Build a fun mini-game where a box moves with arrow keys!

⏰ Duration: 90–120 minutes

Level: Intermediate (students know basic JS and DOM)

📍 Part 1: What Are Events? (10 minutes)

🎯 Learning Goal

Understand what an "event" is and how to use `addEventListener()` to make your page interactive.

🧠 Key Concept

An **event** is something the user does — like:

- Pressing a key on the keyboard
- Clicking a mouse button
- Moving the mouse

You can "listen" for these actions using:

```
document.addEventListener("click", function() {
  alert("You clicked!");
});
```

This means:

🗣️ “When the user **clicks anywhere**, run this function.”

🔍 Real-Life Analogy

> Imagine your webpage is listening like a security camera:
>
> - Someone presses a key = "Beep! I heard that!"
> - Someone clicks = "Click detected!"
>   Events allow your webpage to
>   *react* just like that.

🧪 Quick Try (Live Demo)

Ask students to:

1. Open their browser console
2. Paste and run this:

```
document.addEventListener("click", function() {
  console.log("Page clicked!");
});
```

1. Try clicking anywhere on the page — see the message in the console?

🧠 Teacher Tips

- Ask students: “What are some events you've seen on websites?”
- Reinforce: We *don’t call* the function immediately — it runs *only when the event happens*

> ❗ Common mistake: Writing addEventListener("click", alert("clicked!")) — this runs right away!

Ready to respond to keyboard events? Let’s go! ⌨️

⌨️ Part 2: Keyboard Events (20 minutes)

🎯 Learning Goal

Understand how to listen for keyboard input using `keydown`, and respond to specific key presses.

✅ What is `keydown`?

The `keydown` event runs when **any key** is pressed down.

Example:

```
document.addEventListener("keydown", function(event) {
  console.log("Key pressed:", event.key);
});
```

> This will log the exact key the user presses — like "a", "Enter", "ArrowUp", or " " (spacebar).

💡 Student Tasks

1. **Show a message when spacebar is pressed**

```
document.addEventListener("keydown", function(event) {
  if (event.key === " ") {
    alert("Spacebar pressed!");
  }
});
```

1. **Change background color when "a" is pressed**

```
document.addEventListener("keydown", function(event) {
  if (event.key === "a") {
    document.body.style.backgroundColor = "lightblue";
  }
});
```

🧠 Teacher Tips

- Emphasize: `event.key` gives you the **string name** of the key
- Optional live test: Ask students to press any key and watch the console output
- Let students try changing to other keys like `"w"`, `"Enter"`, `"ArrowDown"`

🔧 Optional Challenge

- Create a typing effect: show every letter the user types inside a `<p>` tag
- Detect a secret code like `Konami Code` (`↑ ↑ ↓ ↓ ← → ← → B A`) for a fun surprise

✅ Recap

Concept	Code Example
Listen to key press	`document.addEventListener("keydown", fn)`
Get which key	`event.key`
React to it	`if (event.key === "a") { ... }`

> Ready to take control of the mouse? 🖱️ Let's go to mouse events!

🖱️ Part 3: Mouse Events (20 minutes)

🎯 Learning Goal

Learn how to detect and respond to mouse actions like clicks, movement, and hovering.

✅ Common Mouse Events

Event Name	Triggered When...
`click`	User clicks an element
`mouseenter`	Mouse moves **into** an element
`mousemove`	Mouse moves **anywhere** on page

🧪 Example: `mouseenter`

```
document.getElementById("box").addEventListener("mouseenter", () ⇒ {
  console.log("Mouse entered the box!");
});
```

> 🔍 This runs once when the mouse enters the box.
>
> Great for changing color, adding glow effects, etc.

💡 Student Tasks

🖱️ Task 1: Show mouse coordinates

```
document.addEventListener("mousemove", function(e) {
  console.log("X:", e.clientX, "Y:", e.clientY);
});
```

> Try this in the console and watch how the coordinates update live!

🎨 Task 2: Change color on hover

Create a box and change its color when the mouse enters:

HTML (example setup)

```
<div id="colorBox" style="width:100px;height:100px;background:gray;"></div
>
```

JS

```
document.getElementById("colorBox").addEventListener("mouseenter", functi
on() {
  this.style.backgroundColor = "orange";
});
```

🧠 Teacher Tips

- Ask students: "Where have you seen hover effects on websites?"
- Show live how `mousemove` floods the console — and suggest throttling later if advanced
- Let students experiment: what happens if you use `mouseleave`?

🧩 Optional Challenges

- Make the box follow the mouse using `.style.left` / `.style.top`
- Show the coordinates **inside** the box instead of console
- Use `click` to toggle background color or visibility

✅ Recap

Mouse Event	Use Case
`click`	Buttons, toggles
`mouseenter`	Hover effects
`mousemove`	Real-time interaction, games

> Next up: let's combine what we've learned to move a box with keys! 🎮

🎮 Part 4: Mini Project — Move the Box! (30–40 minutes)

✨ Project Goal

Create a simple game where a red box moves **around a game area** using:

- Arrow keys (`↑ ↓ ← →`)
- OR `W A S D` keys

This reinforces:

- `keydown` events
- DOM positioning (`.style.left`, `.style.top`)
- Conditionals and variables

✅ Starter Code (HTML)

```html
<div id="gameArea" style="position:relative;width:400px;height:400px;border:1px solid black;">
  <div id="player" style="width:40px;height:40px;background:red;position:absolute;left:0;top:0;"></div>
</div>
```

✅ Starter Code (JavaScript)

```javascript
let box = document.getElementById("player");
let x = 0;
let y = 0;

// Listen for arrow key or WASD key press
document.addEventListener("keydown", function(e) {
  if (e.key === "ArrowRight" || e.key === "d") x += 10;
  if (e.key === "ArrowLeft" || e.key === "a") x -= 10;
  if (e.key === "ArrowUp" || e.key === "w") y -= 10;
  if (e.key === "ArrowDown" || e.key === "s") y += 10;

  // Move the box
  box.style.left = x + "px";
  box.style.top = y + "px";
});
```

🧪 Breakdown for Students

Code Line	What It Does
`let x = 0; let y = 0;`	Stores box's current position
`e.key === "ArrowUp"`	Checks which key is pressed
`x += 10` / `y -= 10`	Changes the position
`box.style.left = x + "px";`	Visually moves the box

🧩 Student Challenges (Optional Enhancements)

1. ❌ **Stay Inside the Game Area**

```
// Prevent moving beyond the boundaries
if (x < 0) x = 0;
if (x > 360) x = 360; // 400 - 40
if (y < 0) y = 0;
if (y > 360) y = 360;
```

2. 🎯 **Count Moves**

```
let moveCount = 0;
moveCount++;
console.log("Moves:", moveCount);
```

3. 💥 **Add Effects**

    - Change box color briefly when it moves
    - Add a movement sound (`Audio` API)
    - Show key name on screen

🧠 Teacher Tips

- Encourage pair programming or group testing (e.g., one presses keys, one logs output)
- Let advanced students turn it into a mini maze or race game
- Ask students:
    - "What happens if you hold the key down?"
    - "Can we make it move faster?"

✅ Recap

Skill	Used In
`keydown`	Detecting movement keys
Positioning	`.style.left`, `.style.top`
Conditional logic	Handling multiple keys

> 🧩 You just built your first interactive game! Try customizing it — make it yours!

🔁 Part 5: Optional Challenges

- Move multiple boxes with different keys
- Make a button follow the mouse using `mousemove`
- Catch a moving box with your mouse

✅ Recap (10 min)

Concept	Code Example
Keyboard Input	`keydown` , `event.key`
Mouse Movement	`mousemove` , `clientX` , `clientY`
DOM Interaction	`element.style.left = ...`

- Events let your page **listen and respond** to users
- Combined with CSS, they bring your page to life!

Next time, we'll explore **page structure and multi-view apps** 🎯