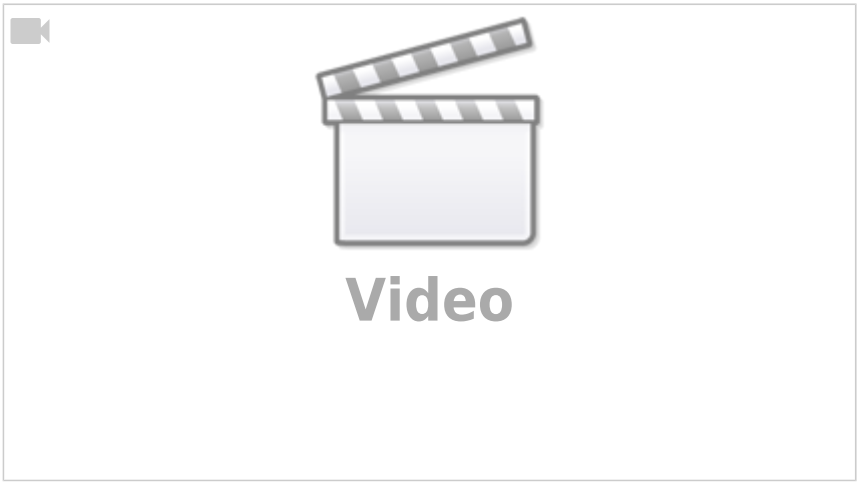
LU04b – Events & State

Events & EventListener

Ein **Event** ist ein Signal, das der Browser aussendet, wenn etwas passiert – ein Klick, eine Tastatureingabe, eine Berührung. Mit einem **EventListener** registrieren Sie eine Funktion, die auf dieses Signal reagiert.

```
const btn = document.querySelector('#mein-button');

btn.addEventListener('click', () => {
  console.log('Geklickt!');
});
```



Video zum Thema Events und Eventlisteners

Verbindung zum Alarado-Projekt: Beim Dark/Light Toggle haben Sie bereits `addEventListener` verwendet: `checkbox.addEventListener('change', toggleTheme)`. Heute nutzen wir dasselbe Muster für das Accordion.

Die wichtigsten Event-Typen

Allgemeine Interaktion

Event	Wann?	Einsatz
`click`	Maustaste oder Tap (mit ~300ms Verzögerung)	Buttons, Links, Toggle
`change`	Wert geändert + Fokus verloren	Checkbox, Select
`input`	Wert ändert sich sofort (jedes Zeichen)	Live-Suche, Zeichenzähler
`submit`	Formular wird abgesendet	Formular-Validierung
`keydown`	Taste gedrückt	Shortcuts, Keyboard-Navigation
`scroll`	Seite wird gescrollt	Sticky Nav, Lazy Loading
`DOMContentLoaded`	HTML vollständig geparst	JS-Initialisierung

```
// Tastatur: Escape schliesst ein Element
document.addEventListener('keydown', (event) => {
  if (event.key === 'Escape') {
    panel.classList.remove('open');
  }
});

// JS erst ausführen, wenn das DOM bereit ist
document.addEventListener('DOMContentLoaded', () => {
  const buttons = document.querySelectorAll('.accordion-
btn');
  // ...
});
```

Pointer Events (moderner Standard)

Pointer Events sind der **aktuelle Standard** für alle Zeigereingaben. Sie decken Maus, Touchscreen und Stift mit einem einzigen Event-System ab – und ersetzen damit die älteren separaten Maus- und Touch-Events.

Event	Wann?
`pointerdown`	Zeiger gedrückt / Finger berührt Bildschirm
`pointermove`	Zeiger bewegt sich
`pointerup`	Zeiger losgelassen / Finger abgehoben
`pointerenter`	Zeiger betritt Element
`pointerleave`	Zeiger verlässt Element
`pointercancel`	Interaktion abgebrochen (z.B. Anruf, Scroll)

Beispiel: Hover und Klick mit Pointer Events

```
const box = document.querySelector('.box');

box.addEventListener('pointerenter', () => {
  box.classList.add('hovered');
});

box.addEventListener('pointerleave', () => {
  box.classList.remove('hovered');
});

box.addEventListener('pointerdown', (event)
=> {
  // Welches Gerät wird verwendet?
  if (event.pointerType === 'touch') {
    console.log('Touch-Eingabe');
  } else if (event.pointerType === 'mouse') {
    console.log('Maus-Eingabe');
  } else if (event.pointerType === 'pen') {
    console.log('Stift, Druck:',
event.pressure);
  }
});
```

Ältere Alternativen: `mousedown` / `mousemove` / `mouseup` funktionieren nur mit der Maus. `touchstart` / `touchmove` / `touchend` nur auf Touchscreens. Pointer Events ersetzen beide – verwenden Sie in neuen Projekten immer Pointer Events.

Eine vertiefte Beschreibung von Events mit interaktiven Code-Beispielen finden Sie hier: Javascript.info – Events

Das Event-Objekt

Alle Handler erhalten automatisch ein **Event-Objekt** als Parameter:

Eigenschaft / Methode	Beschreibung
`event.target`	Das Element, das das Event ausgelöst hat
`event.type`	Name des Events (z.B. `pointerdown`)

Eigenschaft / Methode	Beschreibung
`event.preventDefault()`	Standardverhalten verhindern (z.B. Link nicht öffnen)
`event.stopPropagation()`	Event nicht weiter nach oben weitergeben

```
// Beispiel: Link-Klick abfangen
document.querySelector('a').addEventListener('click',
(event) => {
  event.preventDefault();
  console.log('Navigation verhindert');
});
```

State – Zustand im UI

State (Zustand) ist der aktuelle Stand Ihrer Benutzeroberfläche. Beim Accordion hat jedes Panel zwei mögliche Zustände: **offen** oder **geschlossen**.

Das Muster kennen Sie bereits vom Toggle:

Zustand	Klasse gesetzt?	aria-expanded
geschlossen	`.panel` (kein `.open`)	„false“
offen	`.panel.open`	„true“

State aus dem DOM lesen

Sie können den Zustand direkt aus dem DOM ablesen, ohne eine eigene Variable zu führen:

```
const panel = document.querySelector('.panel');
const btn = document.querySelector('.accordion-btn');

btn.addEventListener('click', () => {
  const istOffen = panel.classList.contains('open');

  if (istOffen) {
    panel.classList.remove('open');
    btn.setAttribute('aria-expanded', 'false');
  } else {
    panel.classList.add('open');
    btn.setAttribute('aria-expanded', 'true');
  }
});
```

Alle anderen Panels schliessen

Bei einem klassischen Accordion soll immer nur ein Panel offen sein:

```
const buttons = document.querySelectorAll('.accordion-btn');

buttons.forEach(btn => {
  btn.addEventListener('click', () => {
    const panel = btn.nextElementSibling;
    const istOffen = panel.classList.contains('open');

    // Alle Panels schliessen
    buttons.forEach(andererBtn => {
andererBtn.nextElementSibling.classList.remove('open');
      andererBtn.setAttribute('aria-expanded', 'false');
    });

    // Dieses Panel öffnen (nur wenn es vorher zu war)
    if (!istOffen) {
      panel.classList.add('open');
      btn.setAttribute('aria-expanded', 'true');
    }
  });
});
```