JavaScript: zdarzenia i walidacja formularza (ściąga)

Trzy sposoby podpięcia obsługi

Sposób	Zapis	Uwaga
atrybut znacznika	`<button onclick="licz()">Licz</button>`	wywołanie z nawiasami
właściwość elementu	`przycisk.onclick = licz;`	bez nawiasów, tylko jedna funkcja
metoda	`przycisk.addEventListener("click", licz);`	nazwa bez `on`, wiele funkcji naraz
odpięcie obsługi	`przycisk.removeEventListener("click", licz);`	ta sama nazwa i ta sama funkcja

Nazwy zdarzeń

Zdarzenie	Kiedy zachodzi
`onclick`, `ondblclick`	pojedyncze i dwukrotne kliknięcie elementu
`onmousedown`, `onmouseup`	wciśnięcie i zwolnienie przycisku myszy
`onmouseover`, `onmouseout`	kursor wchodzi na element i go opuszcza
`onkeydown`, `onkeyup`	klawisz klawiatury naciśnięty i zwolniony
`onsubmit`, `onreset`	wysłanie i wyczyszczenie formularza, na `<form>`
`onchange`, `oninput`	zmiana po wyjściu z pola i znak po znaku
`onfocus`, `onblur`	wejście kursora do pola i wyjście z niego
`onload`	strona z obrazami wczytana, atrybut znacznika `<body>`
`DOMContentLoaded`	kod HTML przetworzony, obrazy jeszcze nie

Obiekt zdarzenia

Zapis	Co daje
`event.type`	nazwa zdarzenia
`event.target`	element, na którym zdarzenie zaszło
`event.key`	znak naciśniętego klawisza
`event.preventDefault()`	blokuje domyślne działanie, na przykład wysyłkę formularza

Mysz i klawiatura w kodzie

```
promocja.addEventListener("mouseover", function () {
  promocja.style.backgroundColor = "Tomato";
});
uwagi.addEventListener("keyup", function () {
  licznik.innerHTML = uwagi.value.length;
});
```

Wynik: tło elementu o `id="promocja"` zmienia się pod kursorem, a w elemencie o `id="licznik"` rośnie liczba znaków wpisanych w polu uwag.

Walidacja formularza

```
<form id="zam" action="zamow.php" method="post" onsubmit="return sprawdz()">
  <input type="text" id="ilosc" name="ilosc">
  <input type="submit" value="Zamów">
</form>
```

```
function sprawdz() {
  const ilosc = Number(document.getElementById("ilosc").value);
  if (isNaN(ilosc) || ilosc < 1 || ilosc > 10) {
    alert("Podaj liczbę od 1 do 10");
    return false;
  }
  return true;
}
```

Wynik: wartość spoza zakresu zatrzymuje wysyłkę i pokazuje okno z komunikatem, poprawna jedzie metodą POST do pliku `zamow.php`. Warstwę pierwszą daje HTML5, sprawdzany PRZED zdarzeniem `submit`: `required`, `min`, `max`, `pattern`, a `novalidate` w znaczniku `<form>` wyłącza to sprawdzanie.

Pułapki

- W `addEventListener` nazwa zdarzenia jest bez przedrostka: `"click"`, a nie `"onclick"`.
- `przycisk.onclick = licz();` z nawiasami wywołuje funkcję od razu, zamiast ją podpiąć.
- Drugie przypisanie do `onclick` kasuje pierwsze, `addEventListener` dokłada obie funkcje.
- `onsubmit="sprawdz()"` bez słowa `return` nie zatrzyma wysyłki, choćby funkcja zwracała `false`.
- Zdarzenie `submit` zachodzi na `<form>`, nie na przycisku: przycisk tylko je wyzwala.
- `value` to zawsze napis, także w polu `type="number"`, więc porównanie liczbowe wymaga `Number`.
- `keydown` zachodzi przed wpisaniem znaku do pola, więc licznik znaków podpina się pod `keyup`.
- Skrypt w `<head>` nie znajdzie jeszcze elementów: plik przed `</body>` albo `DOMContentLoaded`.
- Walidacja u klienta to wygoda, nie zabezpieczenie: te same reguły powtarza skrypt na serwerze.

Egzamin praktyczny: „przycisk uruchamiający skrypt" to `<button onclick="nazwa()">`, a „skrypt wykonywany po załadowaniu strony" to `<body onload="nazwa()">` albo `<script src="skrypt.js"></script>` przed `</body>`; id służy wyszukiwaniu w skrypcie, name wysyłce na serwer.