

Zdarzenia i formularze w JavaScript

Czas: 2 × 45 min

praca indywidualna

Cel: Obsługa zdarzeń `click`, `input`, `change` i `submit`: przeliczanie zamówienia na żywo oraz walidacja formularza w JavaScript z czytelnymi komunikatami pod polami.

Po zajęciach potrafisz:

- podłącza obsługę zdarzeń metodą `addEventListener` do pojedynczych pól i grup elementów
- dobiera zdarzenie do kontrolki (`input` dla pól liczbowych i tekstowych, `change` dla checkboxów, radio i list)
- odczytuje wartości pól, stan `checked` i atrybuty `data-*` oraz zamienia tekst na liczbę
- zatrzymuje domyślne wystąpienie formularza (`preventDefault`) i waliduje dane, wyświetlając komunikaty pod polami
- wyjaśnia, dlaczego walidacja w przeglądarce nie zastępuje sprawdzania danych na serwerze

Sprzęt i materiały

- komputer z Windows 11 i przeglądarką Chrome, Edge lub Firefox
- folder `r1_26_start` z Pliki_do_zadan (`index.html`, `styl.css`, pusty `skrypt.js`)

Oprogramowanie

- **Visual Studio Code** — rozszerzenie Live Server
code.visualstudio.com
- **Przeglądarka z narzędziami F12** — Elements → panel Event Listeners; Console

Przygotowanie przed zajęciami

- Przypomnij sobie z prezentacji 1.17 `addEventListener`, obiekt zdarzenia i `event.preventDefault()`; z zadania 6 — atrybuty walidacji HTML.
- Skopiuj folder `r1_26_start` jako `zad26` do swojego folderu `nazwisko_imie`.

Bezpieczeństwo

- Do testów używaj zmyślonych danych (np. adres w domenie `example.com`, telefon 600 100 200) — nie wpisuj prawdziwych danych swoich ani kolegów.
- Pamiętaj: walidacja w JavaScript ułatwia życie użytkownikowi, ale nie chroni serwera — dane i tak trzeba sprawdzić w PHP (rok 2).

Przebieg zadania

1 Podsumowanie liczone na żywo

Każde pole ilości ma klasę `ilosc` oraz atrybuty `data-nazwa` i `data-cena`; checkboxy dodatków — klasę `dodatek`. Wartości pól są tekstami — przed liczeniem zamień je funkcją `Number()`.

```
const ilosci = document.querySelectorAll(".ilosc");
function przelicz() {
  let suma = 0;
  ilosci.forEach(pole => {
    const ile = Number(pole.value);
    // ...
  });
}
ilosci.forEach(p => p.addEventListener("input", przelicz));
```

1. Uruchom stronę przez Live Server, wpisz w komentarzu `skrypt.js` swoje imię i nazwisko.
2. Z1. Napisz funkcję `przelicz()`, która czyści listę `#lista-pozycji` i dla każdej pozycji z ilością większą od 0 dopisuje wiersz „Pierogi ruskie × 2 = 48.00 zł”, a wartości sumuje. Podłącz ją do zdarzenia `input` wszystkich pól ilości.
3. Z2. Dolicz zaznaczone dodatki (zdarzenie `change`, właściwość `checked`) i dopisz je do listy. Gdy nic nie wybrano, lista ma pokazywać „Koszyk jest pusty”.
4. Z3. Obsłuż wybór odbioru (radio `name="dostawa"`, selektor `input[name=dostawa]:checked`): dostawa kosztuje 9 zł, a od 100 zł zamówienia jest gratis — wtedy w `#koszt-dostawy` wyświetl słowo „gratis”.
5. Z4. Wyświetl łączną kwotę w `#suma` z 2 miejscami po przecinku. Sprawdź: 2 × ruskie, 1 × jagody, śmietana, kompot → 81.00 zł; z dostawą 90.00 zł; po dodaniu 1 × z mięsem → 108.00 zł i dostawa „gratis”.

2 Walidacja przy wysłaniu

Pod każdym polem jest element `span.blad` o id `blad-<id pola>`; klasa `niepoprawne` obramowuje pole na czerwono.

1. Z5. Kliknij „Zamawiam” bez wypełniania danych i zobacz, co się dzieje z adresem strony. Dodaj do znacznika `form` atrybut `novalidate` (walidację przejmie skrypt), a w obsłudze zdarzenia `submit` wywołaj `event.preventDefault()`.
2. Z6. Napisz funkcję `pokazBlad(id, tekst)`, która wpisuje komunikat do `#blad-<id>` i dodaje polu klasę `niepoprawne` (pusty tekst — usuwa ją). Zwraca `true`, gdy pole jest poprawne.
3. Z7. W funkcji `waliduj()` sprawdź: imię i nazwisko co najmniej 3 znaki (po `trim`), telefon dokładnie 9 cyfr (spacje usuń, wzorzec `/^\d{9}$/` i metoda `test`), e-mail ze znakiem `@` i kropką w domenie, wybraną godzinę, zaznaczoną zgodę. Wyświetl **wszystkie** błędy naraz, a nie tylko pierwszy.
4. Z8. Gdy formularz jest niepoprawny lub nie zamówiono żadnych pierogów, wyświetl w `#komunikat` odpowiednią informację i ustaw kursor (`focus()`) w pierwszym błędnym polu. Gdy wszystko jest dobrze — komunikat „Dziękujemy, Anna Nowak! Zamówienie na kwotę 108.00 zł będzie gotowe o 13:00.” (klasa `ok`) i zablokuj przycisk (`disabled`).
5. Z9. Dodaj do pól imienia, telefonu i e-maila obsługę zdarzenia `input`, która od razu usuwa komunikat błędu, gdy użytkownik zaczyna poprawiać pole.

3 Testy i dokumentacja

1. Z10. Przetestuj formularz według tabeli w sprawozdaniu: puste pola, imię „Al”, telefon „12 345 678”, e-mail bez kropki w domenie, brak godziny, brak zgody, zamówienie bez pierogów, zamówienie poprawne.
2. Z11. W F12 → Elements zaznacz przycisk „Zamawiam” lub formularz i obejrzyj panel Event Listeners. Zrób zrzut.
3. Zrób zrzuty: podsumowanie z kwotą 108.00 zł i dostawą gratis, formularz z wszystkimi pięcioma komunikatami błędów, komunikat o przyjęciu zamówienia.

W sprawozdaniu umieść

- ☐ tabela Z1–Z11 z fragmentami kodu i wynikami
- ☐ tabela testów z Z10 (dane wejściowe, oczekiwany i otrzymany wynik)
- ☐ zrzuty: podsumowanie 108.00 zł, pięć komunikatów błędów, zamówienie przyjęte, panel Event Listeners
- ☐ folder zad26 z plikami strony
- ☐ odpowiedzi na pytania kontrolne

Pytania kontrolne

1. Czym różnią się zdarzenia `input` i `change`? Które wybrać dla pola liczbowego, a które dla checkboxa?
2. Co robi `event.preventDefault()` w obsłudze `submit` i co by się stało bez niego?
3. Dlaczego `pole.value` trzeba zamienić funkcją `Number()` przed dodawaniem? Co da `"2" + 1`?
4. Dlaczego dane z formularza trzeba ponownie sprawdzić na serwerze, mimo walidacji w JavaScript?

Kryteria oceny

Kryterium	Punkty
Z1–Z2 — pozycje i dodatki w podsumowaniu, liczby z data-cena, „Koszyk jest pusty”	4
Z3–Z4 — koszt dostawy z progiem 100 zł, poprawne kwoty kontrolne 81 / 90 / 108 zł	3
Z5–Z7 — <code>preventDefault</code> , funkcja <code>pokazBład</code> , 5 reguł walidacji, wszystkie błędy naraz	5
Z8–Z9 — komunikaty końcowe, focus na pierwszym błędzie, blokada przycisku, czyszczenie błędu przy poprawianiu	3
Z10–Z11 — tabela testów i zrzuty	3
Odpowiedzi na pytania kontrolne	2
Razem	20

Sprawozdanie przygotuj na wzorze dołączonym do zadania (plik .docx o tym samym numerze).