

Flexbox w praktyce

Czas: 2 × 45 min

praca indywidualna

Cel: Budowanie typowych fragmentów układu strony Flexboxem: poziomego paska nawigacji, wyśrodkowanego bloku, sekcji kart z zawijaniem i stopki przyklejonej do dołu okna.

Po zajęciach potrafisz:

- tworzy kontener flex i ustawia elementy na osi głównej (justify-content) i poprzecznej (align-items)
- buduje poziome menu z odstępami gap i rozsuwa elementy nagłówka
- wyśrodkowuje blok w pionie i poziomie
- układa karty z zawijaniem (flex-wrap) i steruje ich szerokością skrótem flex
- przykleja stopkę do dołu okna przy krótkiej treści strony

Sprzęt i materiały

- komputer z Windows 11 i przeglądarką Chrome, Edge lub Firefox
- folder `r1_13_start` z Pliki_do_zadan (index.html i style.css)

Oprogramowanie

- **Visual Studio Code z rozszerzeniem Live Server** — Live Server: Extensions (Ctrl+Shift+X) → „Live Server” code.visualstudio.com
- **Flexbox Froggy (opcjonalnie)** — gra w przeglądarce, wersja polska flexboxfroggy.com/#pl
- **Narzędzia F12 przeglądarki** — plakietka „flex” przy elemencie w zakładce Elements włącza nakładkę Flexbox developer.chrome.com/docs/devtools/css/flexbox

Przygotowanie przed zajęciami

- Przypomnij sobie z wykładu 1.10 pojęcia: kontener, elementy flex, oś główna i poprzeczna.
- Skopiuj folder `r1_13_start` do folderu `nazwisko_imie` jako `zad13`.

Bezpieczeństwo

- Pracuj na kopii plików; nie zmieniaj pliku index.html — całe zadanie wykonujesz w CSS.

Przebieg zadania

1 Rozgrzewka (opcjonalnie, maks. 10 min)

Jeśli skończysz zadanie wcześniej, wróć do gry i przejdź dalsze poziomy.

1. Otwórz Flexbox Froggy i przejdź poziomy 1–10. Zanotuj numer ostatniego ukończonego poziomu.
2. Uruchom `zad13/index.html` przez Live Server. Zrób zrzut strony przed zmianami — porównasz go z wynikiem końcowym.

2 Pasek nawigacji

Reguły dopisuj w `style.css` pod komentarzem `Zadanie 13`. Po każdej zmianie zaznacz kontener w F12 i kliknij plakietkę `flex`, aby zobaczyć nakładkę.

- F1. Zamień `.naglowek` w kontener flex. Logo, menu i przycisk mają stać w jednym rzędzie i być wyśrodkowane w pionie (`align-items`).
- F2. Rozsuń elementy nagłówka: logo przy lewej krawędzi, przycisk „Zapisz się” przy prawej (`justify-content: space-between`).
- F3. Ustaw odnośniki listy `.menu` w poziomie z odstępem 8px (`display: flex` i `gap`), bez użycia `float` i `inline-block`.
- F4. Przesuń menu do prawej, tuż obok przycisku: nadaj elementowi `nav` `margin-left: auto`, a nagłówkowi `gap: 16px`. Wyjaśnij w sprawozdaniu, dlaczego automatyczny margines „wypycha” element.

3 Centrowanie

- F5. Wyśrodkuj biały blok `.hero-tekst` w sekcji `.hero` (wysokość 260px) jednocześnie w poziomie i w pionie — tylko właściwościami kontenera flex.
- F6. Zmień na chwilę `flex-direction` sekcji hero na `column` i sprawdź, czy blok nadal jest wyśrodkowany. Zapisz w sprawozdaniu, która właściwość odpowiada teraz za pion, a która za poziom. Przywróć `row`.

4 Karty z zawijaniem

Skrót `flex: 1 1 260px` oznacza: element może rosnąć (grow 1), może się kurczyć (shrink 1), a jego szerokość bazowa to 260px.

- F7. Ułóż karty w kontenerze `.karty` w rzędzie z odstępem 16px. Sprawdź, co się dzieje przy szerokości okna 600px (tryb urządzeń Ctrl+Shift+M). Następnie włącz zawijanie (`flex-wrap: wrap`).
- F8. Nadaj kartom `flex: 1 1 260px`. Zmieniaj szerokość okna od 1280px do 375px i zapisz, przy jakich szerokościach w rzędzie są 4, 3, 2 i 1 karta.
- F9. Wyrównaj dolną linię kart: każda `.karta` ma być kontenerem flex w kolumnie, a akapit `.info` ma się trzymać dołu karty (`margin-top: auto`). Wszystkie linie z liczbą graczy w jednym rzędzie mają być na tej samej wysokości.
- F10. Bez zmiany HTML przenieś ostatnią kartę („Labirynt Smoka”) na początek właściwością `order` i wyróżnij ją pomarańczowym obramowaniem 2px (`#f59e0b`).

5 Stopka przyklejona do dołu

Przy krótkiej treści stopka „wisi” w połowie okna. Aby to zobaczyć, ustaw w trybie urządzeń wysokość okna 1400px (np. 1280 × 1400).

- F11. Zamień `body` w kontener flex w kolumnie o minimalnej wysokości `100vh` i pozwól `main` zająć całe wolne miejsce (`flex: 1`). Stopka ma dotykać dolnej krawędzi okna.
- F12. Ułóż dwa akapity stopki w jednym rzędzie: tekst o klubie po lewej, opiekun po prawej; na wąskim ekranie akapity mają się zawinąć jeden pod drugi.
- F13. Zrób zrzuty wyniku przy 1280 × 1400 i 375 × 800. Sprawdź `style.css` walidatorem CSS W3C (<https://jigsaw.w3.org/css-validator/>).

W sprawozdaniu umieść

- ☐ tabela F1–F13 z regułami CSS i opisem wyniku
- ☐ zrzuty: strona przed zmianami, po zmianach przy 1280 × 1400 i przy 375 × 800, nakładka Flexbox z F12 na kontenerze `.karty`
- ☐ tabela szerokości okna i liczby kart w rzędzie (F8)

- ☐ wyjaśnienie działania margin-left: auto (F4) i zamiany osi (F6)
- ☐ folder `zad13` z plikami
- ☐ odpowiedzi na pytania kontrolne

Pytania kontrolne

1. Która właściwość wyrównuje elementy na osi głównej, a która na poprzecznej? Co się zmienia po ustawieniu flex-direction: column?
2. Co oznaczają trzy wartości w skrócie flex: 1 1 260px?
3. Dlaczego właściwość order nie powinna zmieniać kolejności istotnej dla treści (pomyśl o czytniku ekranu i klawiszu Tab)?
4. Jak działa stopka przyklejona do dołu: która reguła powoduje, że main „rozpycha się” na wolne miejsce?

Kryteria oceny

Kryterium	Punkty
Nagłówek: logo po lewej, menu (4 odnośniki w rzędzie) i przycisk po prawej, wyśrodkowane w pionie (F1–F4)	4
Blok hero wyśrodkowany w pionie i poziomie, opis zamiany osi (F5–F6)	3
Karty zawijają się, flex 1 1 260px, tabela szerokości, linia .info wyrównana, Labirynt Smoka pierwszy z obramowaniem (F7–F10)	5
Stopka przy dolnej krawędzi okna 1280 × 1400, dwa akapity w rzędzie (F11–F12)	3
Zrzuty, walidacja CSS bez błędów, sprawozdanie	2
Odpowiedzi na pytania kontrolne	3
Razem	20

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