

Podstawy informatyki
HTML5 / CSS3

Laboratorium 9

Flexbox i pozycjonowanie CSS

Czas trwania: 2 × 45 min
Tryb: studia stacjonarne
Jednostka: Katedra Elektrotechniki i Technologii Nadprzewodnikowych
Prowadzący: dr inż. Michał Łanczont

Cel ćwiczenia

Celem jest opanowanie Flexboxa jako jednowymiarowego modelu układu oraz pozycjonowania **relative**, **absolute**, **sticky** i **fixed**. Student ma rozumieć osie, dystrybucję wolnej przestrzeni, bazowy rozmiar elementu oraz wpływ kolejności wizualnej na dostępność.

Wstęp

Flexbox jest jednowymiarowym modelem układu: najważniejsze jest rozmieszczanie elementów wzdłuż osi głównej, z dodatkowymi mechanizmami wyrównania w osi poprzecznej. W drugiej części ćwiczenia analizowane jest pozycjonowanie przez **position**.

Kontener flex

Właściwość	Znaczenie
<code>display: flex</code>	tworzy kontener flex
<code>flex-direction</code>	określa kierunek osi głównej: row, column oraz warianty reverse
<code>flex-wrap</code>	pozwala zawijać elementy na kolejne linie
<code>justify-content</code>	rozmieszcza wolną przestrzeń w osi głównej
<code>align-items</code>	wyrównuje elementy w osi poprzecznej
<code>align-content</code>	rozmieszcza wiele linii flex, gdy istnieje wolna przestrzeń w osi poprzecznej
<code>gap</code>	odstęp pomiędzy elementami/liniami

```
.container-center {  
  display: flex;  
  justify-content: center;  
  align-items: center;  
  gap: 1rem;  
  min-height: 300px;  
}
```

Element flex

Właściwość	Znaczenie
<code>order</code>	wizualna kolejność elementu
<code>flex-grow</code>	udział we wzroście przy wolnej przestrzeni
<code>flex-shrink</code>	udział w kurczeniu przy braku miejsca
<code>flex-basis</code>	rozmiar bazowy przed rozdzieleniem wolnej przestrzeni
<code>align-self</code>	indywidualne wyrównanie w osi poprzecznej

Ważna korekta skrótu **flex**: jednoargumentowe **flex: 1** rozwija się praktycznie do **flex: 1 1 0%**, a nie do **1 1 auto**. Jeżeli potrzebny jest bazowy rozmiar wynikający z **width** lub zawartości, można użyć **flex: 1 1 auto** jawnie.

```
.item-a { flex: 1 1 0%; }  
.item-b { flex: 2 1 0%; }  
.item-c { flex: 0 0 200px; }
```

order i dostępność

order zmienia kolejność wizualną, ale nie porządek elementów w DOM i zwykle nie kolejność odczytu technologii asystujących czy nawigacji klawiaturą. Nie należy używać go do naprawiania błędnej logicznej kolejności HTML.

position i koordynaty

Dla elementów innych niż **static** można wykorzystywać właściwości **top**, **right**, **bottom**, **left** lub współczesny skrót logiczny **inset**.

Wartość	Zachowanie
relative	element pozostaje w normalnym układzie; przesunięcie jest liczone od jego pierwotnej pozycji; może ustanowić punkt odniesienia dla dziecka absolute
absolute	element jest wyjęty z normalnego przepływu i pozycjonowany względem odpowiedniego containing block
fixed	zwykle pozycjonowany względem viewportu i pozostaje w miejscu podczas przewijania
sticky	zachowuje się jak element w normalnym przepływie, a po osiągnięciu progu “przykleja się” w obrębie kontenera przewijania

Relative + absolute

```
.karta-produktu {  
  position: relative;  
  width: 300px;  
  min-height: 200px;  
}  
  
.znak-promocja {  
  position: absolute;  
  top: 0;  
  right: 0;  
  padding: 5px 10px;  
  background: #e67e22;  
  color: white;  
}
```

Najbliższy odpowiednio pozycjonowany przodek jest typowym punktem odniesienia dla **absolute**. W zaawansowanym CSS również inne właściwości mogą tworzyć containing block, ale na tym etapie najważniejszy jest wzorzec **parent relative / child absolute**.

Sticky

```
.menu {  
  position: sticky;  
  top: 0;  
  z-index: 10;  
  background: white;
```

```
}
```

Aby sticky było widoczne, musi istnieć przestrzeń do przewijania. Zachowanie może być ograniczone przez najbliższy kontener przewijania i jego właściwości `overflow`.

Przykład kompletny – trzy karty Flexbox

```
.cards {
  display: flex;
  flex-wrap: wrap;
  gap: 1rem;
  align-items: stretch;
}

.card {
  flex: 1 1 16rem;
  position: relative;
  padding: 1rem;
  border: 1px solid #ccc;
}

.card.featured {
  flex-grow: 2;
}

.badge {
  position: absolute;
  top: 0.5rem;
  right: 0.5rem;
}
```

Wartość `flex-basis: 16rem` pozwala elementom zawijać się, gdy wiersz robi się zbyt wąski. Jest to częsty wzorzec prostego responsywnego układu bez konkretnego breakpointu.

Typowe błędy Flexbox i position

- mylenie osi głównej i poprzecznej po zmianie `flex-direction`,
- używanie `align-content` przy pojedynczej linii flex,
- oczekiwanie, że `flex: 1` oznacza `1 1 auto`,
- pozycjonowanie dziecka absolute bez świadomego punktu odniesienia,
- używanie `order` do logicznego przedstawiania treści,
- sticky bez ustawionego progu, np. `top: 0`.

Przebieg ćwiczenia

Zadanie 1. Osie, gap i zawijanie

Do wykonania

Utwórz `.galeria` z pięcioma elementami. Przetestuj `flex-direction: row/column`, `justify-content`, `align-items`, `gap` i `flex-wrap`. Dla każdej zmiany wskaż oś główną i poprzeczną.

Zadanie 2. grow, shrink i basis

Do wykonania

Utwórz trzy elementy bez `height: 100%`. Pierwszy ustaw `flex-grow: 1`, drugi `flex-grow: 2`, trzeci `flex: 0 0 200px`. Zmniejsz kontener i obserwuj rozdział wolnej przestrzeni. Następnie włącz/wyłącz `flex-shrink` w trzecim elemencie.

Zadanie 3. align-self i order

Do wykonania

Ustaw kontenerowi `align-items: center` i wysokość 300px. Pierwszemu elementowi nadaj `align-self: flex-end`. Drugiemu nadaj `order: 10`. Porównaj kolejność wizualną z kolejnością HTML i sprawdź Tab/odczyt kodu źródłowego.

Zadanie 4. Sticky menu

Do wykonania

Utwórz nagłówek, menu i treść o wysokości wystarczającej do przewijania. Menu ustaw `position: sticky; top: 0`. Dodaj tło i `z-index`, aby treść nie prześwitywała. Sprawdź, w którym momencie element zaczyna się kleić.

Zadanie 5. Nakładka absolute

Do wykonania

Utwórz `.obraz-kontener` z `position: relative` i obrazem. Wewnątrz dodaj `.napis-rog` z `position: absolute; bottom: 0; right: 0`. Dodaj półprzezroczyste tło i padding. Usuń na chwilę `position: relative` z rodzica i wyjaśnij zmianę punktu odniesienia.

Zadanie 6. Fixed kontra sticky

Do wykonania

Przygotuj mały przycisk/znacznik testowy z `position: fixed; bottom: 1rem; right: 1rem`. Porównaj jego zachowanie podczas przewijania z menu sticky.

Zadanie końcowe

Integracja

Zbuduj responsywny pasek kart produktu wykorzystujący Flexbox. Jedna karta ma mieć odznakę pozycjonowaną absolutnie, a nad kartami ma znajdować się sticky menu. Kolejność HTML ma pozostać logiczna bez polegania na `order`.

Kryteria zaliczenia

- rozumienie osi Flexbox,
- poprawne użycie grow/shrink/basis i skrótu flex,
- świadome użycie order bez psucia kolejności logicznej,

- działające sticky i absolute z właściwym punktem odniesienia,
- brak sprzecznych ustawień uniemożliwiających demonstrację align-self.

Sprawozdanie

Do odbioru należy przygotować pliki źródłowe utworzone podczas ćwiczenia. Jeżeli prowadzący wymaga sprawozdania, powinno ono zawierać: krótki opis rozwiązania, najważniejsze fragmenty kodu, zrzut ekranu wyniku oraz wnioski dotyczące napotkanych problemów. Pliki HTML i CSS należy zachować w strukturze katalogów zgodnej z instrukcją.

Materiały pomocnicze

- HTML Living Standard: <https://html.spec.whatwg.org/>
- MDN Web Docs – HTML: <https://developer.mozilla.org/docs/Web/HTML>
- MDN Web Docs – CSS: <https://developer.mozilla.org/docs/Web/CSS>
- Walidator HTML: <https://validator.w3.org/>