infoShare ACADEMY

CSS - Animacje

infoShare Academy

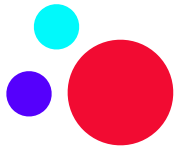
infoShareAcademy.com

HELLO

Jakub Wojtach

Senior **full stack** developer

infoShare ACADEMY



Rozgrzewka i agenda

Zacznijmy ten dzień z przytupem!

Rozgrzewka

Big Bang

Boskie stworzenie ziemii

Rozgrzewka

Własny dom z ogrodem

Penthouse

Rozgrzewka

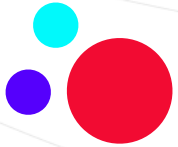
Hip hop

Rock

Rozgrzewka

Stu dobrych kolegów i koleżanek

Dwoje przyjaciół



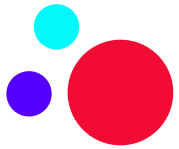
Agenda

- Dokumentacja
- Transformacje
- Przejścia
- Animacje

infoShare ACADEMY

Współpraca

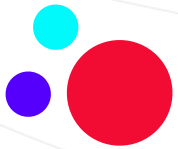
- Zadajemy pytania w dowolnym momencie - kanał **merytoryka**
- Krótkie przerwy (**5 min**) co godzinę
- Długa przerwa (**20 min**) po ostatnim bloku



Zdobywanie wiedzy

I jej poszerzanie

infoShareAcademy.com

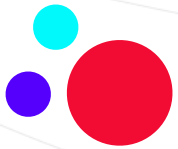


Dobre źródła

- https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Animations/Using_CSS_animations
- https://www.w3schools.com/css/css3_animations.asp
- https://flaviocopes.com/css-transitions/

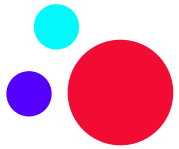
Tytułem wprowadzenia

infoShare ACADEMY



Wpływ developera na UX

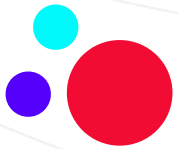
- Wielokrotnie zdarzało mi się przepychać zmiany, które UI przygotował celem poprawienia UX
- Nie należy bać się przedstawiać swojej opinii
- Jeśli widzieliśmy lepsze rozwiązanie - warto próbować przekazać je designerowi



Transformacje

I ich praktyczne wykorzystanie

infoShareAcademy.com



Przekształcenie scaleX

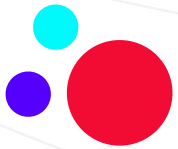
- Pozwala na zmianę szerokości elementu.

Zwiększenie szerokości dwukrotnie

```
div {
  transform: scaleX(2);
}
```

Zmniejszenie szerokości dwukrotnie

```
div {
  transform: scaleX(0.5);
}
```



Przekształcenie scaleY

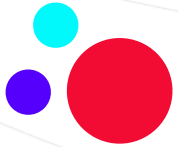
- Pozwala na zmianę wysokości elementu.

Zwiększenie wysokości trzykrotnie

```
div {
  transform: scaleY(3);
}
```

Zmniejszenie wysokości pięciokrotnie

```
div {
  transform: scaleY(0.2);
}
```



Przekształcenie scale

- Pozwala na zmianę wysokości i szerokości elementu.
- Jest to shorthand.
- W momencie gdy podamy jedynie jedną wartość - zostanie ona zastosowana zarówno do szerokości jak i wysokości.

Zwiększenie szerokości dwukrotnie, a wysokości trzykrotnie

```
div {
  transform: scale(2, 3);
}
```

Zmniejszenie szerokości i wysokości dwukrotnie

```
div {
  transform: scale(0.5);
}
```

Obracanie rotateX

- Pozwala na obrót elementu względem osi X bez deformowania go.

```
div {
  transform: rotateX(3.142rad);
}

div {
  transform: rotateX(45deg);
}

div {
  transform: rotateX(-0.2turn);
}
```

`rotateX(a)` jest równoważne przekształceniu **`rotate3d(1, 0, 0, a)`**. Przy czym parametr a oznacza angle

Wizualne przykłady:
https://developer.mozilla.org/en-US/docs/Web/CSS/transform-function/rotateX

Obracanie rotateY

- Pozwala na obrót elementu względem osi Y bez deformowania go.

```
div {
  transform: rotateY(3.142rad);
}

div {
  transform: rotateY(45deg);
}

div {
  transform: rotateY(-0.2turn);
}
```

`rotateY(a)` jest równoważne przekształceniu **`rotate3d(0, 1, 0, a)`**. Przy czym parametr a oznacza angle

Wizualne przykłady:
https://developer.mozilla.org/en-US/docs/Web/CSS/transform-function/rotateY

Obracanie rotate

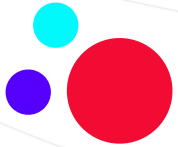
- Pozwala na obrót elementu w przestrzeni 2d bez deformowania go.

```
div {
  transform: rotate(3.142rad);
}

div {
  transform: rotate(45deg);
}

div {
  transform: rotate(-0.2turn);
}
```

Wizualne przykłady:
https://developer.mozilla.org/en-US/docs/Web/CSS/transform-function/rotate



Pochylanie skew

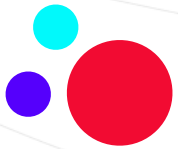
- Pozwala na pochylenie elementu w przestrzeni 2d z deformacją

```
div {
  transform: skew(15deg, 15deg);
}

div {
  transform: rotate(-0.06turn, 18deg);
}

div {
  transform: rotate(-0.2turn);
}
```

Wizualne przykłady:
https://developer.mozilla.org/en-US/docs/Web/CSS/transform-function/skew



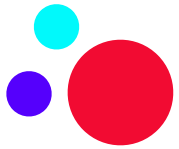
Transform-origin

- Pozwala nam określić pozycję początkową punktu zero interesującego nas elementu HTML
- Daje możliwość ustalenia punktu, wokół którego następuje transformacja (np. Obrót, skalowanie czy pochylenie)
- W zależności od transformacji akcja następuje z innego punktu.

Przykłady:
http://webmaster.helion.pl/index.php/css3-transformacje/6/476-css3-transform-origin

Zadania

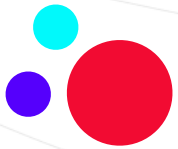
infoShareAcademy.com



Przejścia

I ich praktyczne wykorzystanie

infoShareAcademy.com



Transition-property

- Pozwala na ustawienie efektu przejścia na jaki ma być nałożone płynne przejście.
- Lista wszystkich animowalnych parametrów

```
div {
  transition-property: none
}

div {
  transition-property: font-size;
}
```

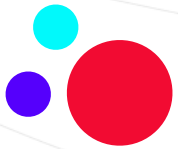
Przykłady:
https://developer.mozilla.org/en-US/docs/Web/CSS/transition-property

Transition-duration

- Pozwala na ustawienie czasu wykonywania animacji do jej zakończenia. Domyślnie jest to 0s, czyli animacja nie nastąpi wcale, a efekt końcowy wykona się **natychmiast**.
- Można ustawić kilka czasów trwania, każde będzie zastosowane do kolejnej wartości wymienionej w kwerendzie **transform-property.**

```
div {
  transition-duration: 500ms.
  transition-property: font-size;
}
```

Przykłady:
https://developer.mozilla.org/en-US/docs/Web/CSS/transition-duration



Transition-delay

- Pozwala na ustalenie jak długo czekamy zanim zastosowany zostanie efekt przejścia ustawiony wcześniej.

```
div {
  transition-delay: 350ms, 500ms;
  transition-property: margin-top, background-color;
  transition-duration: 200ms, 250ms;
}

div {
  transition-delay: 250ms, 300ms;
  transition-property: margin-right, color;
  transition-duration: 300ms, 350ms;
}
```

Przykłady:
https://developer.mozilla.org/en-US/docs/Web/CSS/transition-delay

Transition-timing

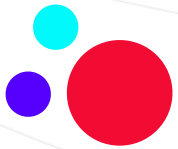
- Pozwala na ustawienie w jaki sposób wartości są obliczane i przekazywane do efektu przejścia.
- Wartość przekazywana to typ funkcji przejścia

```
div {
  transition-timing-function: linear;
}

div {
  transition-timing-function: ease-in;
}

div {
  transition-timing-function: cubic-bezier(.29, 1.01, 1, -0.5);
}
```

Przykłady:
https://developer.mozilla.org/en-US/docs/Web/CSS/transition-timing-function

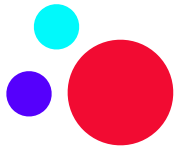


Cubic bezier

- Jeden z algorytmów, który możemy zastosować przy funkcji przejścia.
- Składa się z 4 punktów (p0-p4).
- P0 I P4 to początek i koniec krzywej, a punkty pomiędzy pozwalają na różnorakie wariacje z tym związane.

```
div {
  transition-timing-function: cubic-bezier(.18,.55,.99,.81);
}
```

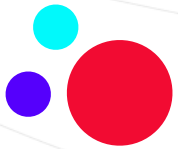
Generator I demo:
https://cubic-bezier.com/#.17,.67,.83,.67



Animacje

I ich praktyczne wykorzystanie

infoShareAcademy.com

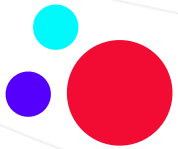


@keyframes

- Umożliwia kontrolę nad przebiegiem animacji. Pozwala na większą kontrolę kolejnych kroków animacji i na tworzenie bardziej wysublimowanych efektów.

```
@keyframes animacja {
  0% { wartość }
  20% { wartość }
  34% { wartość }
  70% { wartość }
  100% { wartość }
}

@keyframes animacja {
  0% { margin-left: 0; background: red; }
  20% { margin-left: 5%; background: blue; }
  34% { margin-left: 15%; background: yellow; }
  70% { margin-left: 8%; background: black; }
  100% { margin-left: 0; background: orange; }
}
```



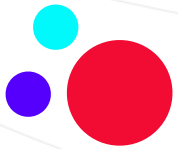
@keyframes

- Umożliwia kontrolę nad przebiegiem animacji. Pozwala na większą kontrolę kolejnych kroków animacji i na tworzenie bardziej wysublimowanych efektów.

```
@keyframes animacja {
  0% { wartość }
  20% { wartość }
  34% { wartość }
  70% { wartość }
  100% { wartość }
}
```

```
@keyframes animacja {
  0% { margin-left: 0; background: red; }
  20% { margin-left: 5%; background: blue; }
  34% { margin-left: 15%; background: yellow; }
  70% { margin-left: 8%; background: black; }
  100% { margin-left: 0; background: orange; }
}
```

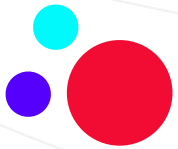
Trochę przykładów
https://www.joshwcomeau.com/animation/keyframe-animations/



Animation-name

- Pozwala ustawić nazwę animacji, która będzie wykorzystana później wraz z propsim keyframes.

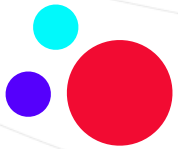
```
animation-name: none;
animation-name: slideIn;
animation-name: sliding-vertically;
```



Animation-duration

- Umożliwia kontrolę nad ustawieniem tego jak długo będzie wykonywany pełen przebieg animacji w jednym cyklu.

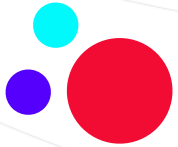
```
animation-duration: 6s;
animation-duration: 1000ms;
animation-duration: 3.64s;
```



Animation-timing-function

- Podobnie jak to było w przypadku transition - jest to funkcja, która jest wykorzystywana do tego, by zobrazować w jaki sposób animacja przebiega w każdym cyklu.

```
animation-timing-function: linear;
animation-timing-function: ease-in;
animation-timing-function: cubic-bezier(0.1, 0.7, 1, 0.1);
```



Animation-play-state

- Umożliwia kontrolę nad ustaleniem, czy animacja ma być uruchomiona, lub zatrzymana.
- Może przyjąć wartości **running** | **paused**
- Stosowane np. gdy chcemy zastosować konkretną animację na hoverze ale nie na domyślnym stanie

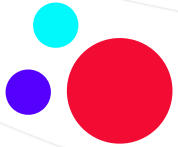
```
animation-play-state: running;
animation-play-state: paused;
```

Przykład
https://developer.mozilla.org/en-US/docs/Web/CSS/animation-play-state

Animation-direction

- Umożliwia kontrolę nad kierunkiem w jakim mają być wykonywane kolejne kroki zapisane w **keyframes** zadeklarowanej animacji.
- Możliwe wartości **normal** | **reverse** | **alternate** | **alternate-reverse**
- **Normal** - po zakończeniu animacja powraca do stanu początkowego i odtwarza się od początku. Wartość domyślna.
- **Reverse** - kierunek odwrotny do normal, wykonywana od końca do początku.
- **Alternate** - w co drugim wykonaniu animacja posiada odwrotny kierunek.
- **Alternate-reverse** - odwrotny kierunek wykonywania i odwrotny co drugie powtórzenie

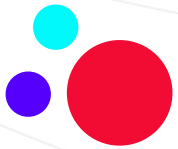
Przykład
https://webkod.pl/kurs-css/wlasciwosci/animacja/animation-direction



Animation-fill-mode

- Umożliwia kontrolę nad tym w jaki sposób animacja css będzie nakładała kolejne wartości na element przed i po wykonaniu
- Wartości **none** | **forwards** | **backwards** | **both**
- **Forwards** - zanim się zacznie
- **Backwards** - kiedy się zakończy
- **Both** - obydwa

Przykład
https://www.w3schools.com/cssref/css3_pr_animation-fill-mode.asp

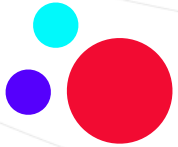


Animation-iteration-count

- Umożliwia kontrolę nad ilością powtórzeń zanim animacja się wykona.

```
animation-iteration-count: 0;
animation-iteration-count: 2;
animation-iteration-count: 1.5;
```

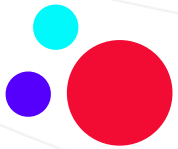
Przykład
https://developer.mozilla.org/en-US/docs/Web/CSS/animation-iteration-count



Animation-delay

- Umożliwia kontrolę nad ilością czasu, który musi nastąpić przed wykonaniem się animacji - zanim animacja zacznie się odtwarzać.
- Przyjmuje wartości numeryczne czasowe, z zasadami:
- Gdy wartość jest **dodatnia** - animacja odtworzy się od początku z opóźnieniem podanym jako wartość
- Gdy wartość jest **ujemna** - animacja odtworzy się **od momentu** podanego jako wartość bez żadnego opóźnienia

```
animation-delay: 1s;
animation-delay: -1s;
```



Animation shorthand

- Umożliwia zapis animacji w sposób skrócony, celem uniknięcia zapisu w wielu wierszach.

Jest to zapis skrótowy dla animation-name, animation-duration, animation-timing-function, animation-delay, animation-iteration-count, animation-direction, animation-fill-mode, oraz animation-play-state.

```
/* @keyframes duration | easing-function | delay |
iteration-count | direction | fill-mode | play-state | name */
animation: 3s ease-in 1s 2 reverse both paused slideIn;

/* @keyframes duration | easing-function | delay | name */
animation: 3s linear 1s slideIn;
```

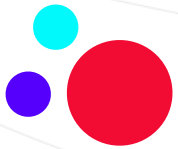
Optymalizacja

InfoShare ACADEMY

Zadanie

Pytania

infoShareAcademy.com



Linki dla chętnych

- https://web.dev/learn/css/animations/
- https://cubic-bezier.com/
- https://medium.com/outsystems-experts/how-to-achieve-60-fps-animations-with-css3-db7b98610108
- https://csstriggers.com/
- https://huijing.github.io/slides/03-css-triggers/#/2
- https://www.joshwcomeau.com/animation/keyframe-animations/

Dziękuję za uwagę

Jakub Wojtach