



Front End 4 Beginners

Be aWWWesome - animacje w CSSie

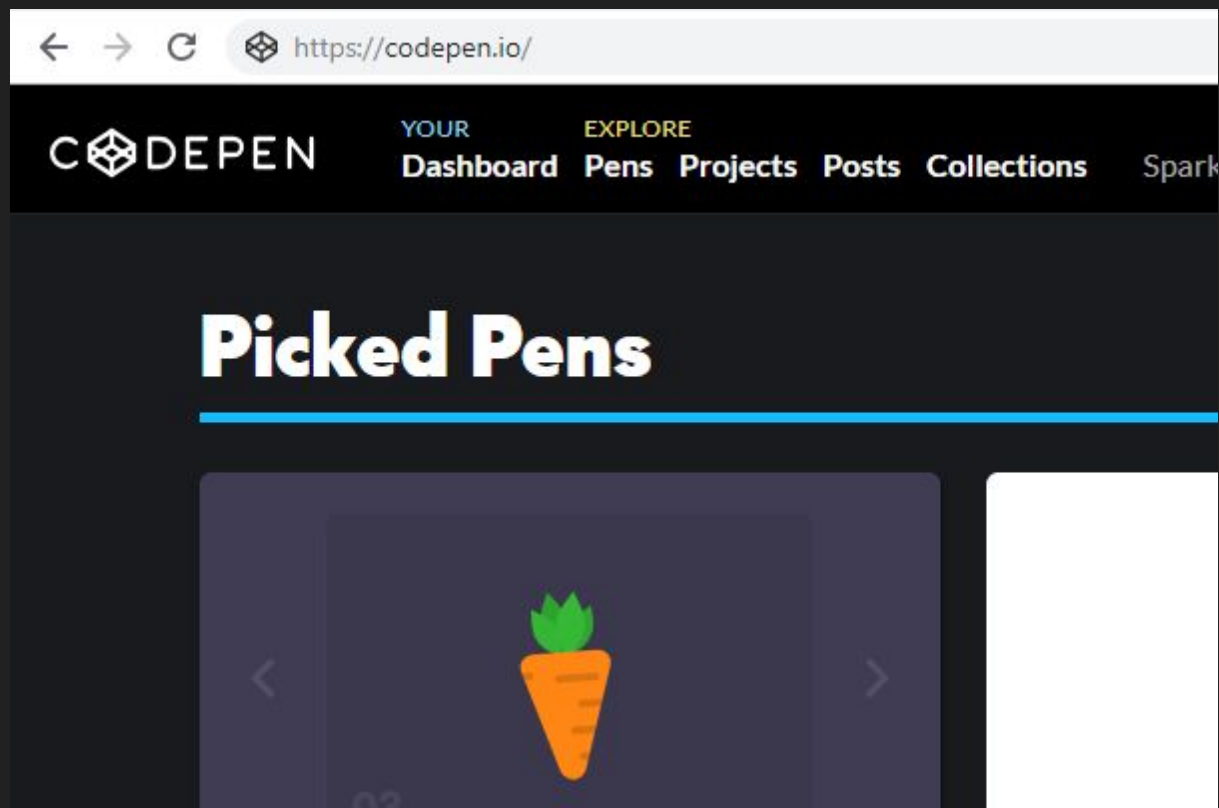
Lista obecności

813641

<https://daftacademy.pl/courses/LGQtQZ>

Animacje w CSS

CodePen.io



Pseudo-klasy

Mogą być dynamiczne w tym sensie, że element "nabywa" lub "traci" pseudo-klasę podczas interakcji z użytkownikiem.

Przykładem jest podświetlenie elementu po wskazaniu go myszką przez użytkownika.



```
.container {  
  width: 40%;  
  height: 80px;  
  padding: 15px;  
  font-size: 1.2em;  
  border: 3px solid #000;  
}  
  
.container:hover {  
  width: 80%;  
  background-color: red;  
}
```

Codepen: <https://codepen.io/dzienisz/pen/ZmZZaw>



Pseudoklasy dynamiczne

Pseudoklasy linkow: :link, :visited

Pseudoklasy akcji uzytkownika: :active, :hover, :focus

Pseudoklasa etykiety: :target

Pseudoklasa jezyka: :lang()

Pseudoklasy interfejsu uzytkownika:

:enabled, :disabled

:checked

Pseudoklasy strukturalne:

:root

:nth-child(), :nth-last-child(), :nth-of-type(), :nth-last-of-type()

:first-child

:last-child

:only-child

:first-of-type, :last-of-type, :only-of-type

:empty

Pseudoklasa negacji: :not()

Codepen: <https://codepen.io/dzienisz/pen/GwLboa>

Pseudo-elementy

Pseudo-elementy pozwalają wstawiać do dokumentu fikcyjne elementy.

Istnieją 4 pseudo-elementy.



```
::first-line  
::first-letter  
::before  
::after
```

Codepen: <https://codepen.io/dzienisz/pen/KrLPYd>

Inne pseudo-elementy

Pseudo-elementy pozwalają wstawiać do dokumentu fikcyjne elementy.

Istnieją 4 pseudo-elementy.



```
::selection {  
  background: hotpink;  
}
```

Codepen: <https://codepen.io/dzienisz/pen/vQwYXP>



Właściwość:
transition

Właściwość: transition

Zauważyć możemy zmianę koloru tła linku po najechaniu kursorem. Zawdzięczamy to pseudoklasie **:hover**. Zmiana koloru jest nagła.

Możemy nadać jej bardziej 'delikatnego' charakteru, dodając właściwość **transition**, czyli płynnego przejścia w czasie między stanami zdefiniowanymi w CSS.



```
a.link {  
  margin: 50px;  
  padding: 10px 20px;  
  background-color: green;  
  color: #fff;  
  /*dodanie funkcji przejścia*/  
  transition: all 0.7s ease-in-out;  
}  
  
a.link:hover {  
  background-color: purple;  
}
```

Codepen: <https://codepen.io/dzienisz/pen/QJRWZv>



```
a.link {  
  margin: 50px;  
  padding: 10px 20px;  
  background-color: green;  
  color: #fff;  
  /*dodanie funkcji przejścia*/  
  transition-property: all;  
  transition-duration: 0.7s;  
  transition-timing-function: ease-in-out;  
  transition-delay: 0s;  
}  
  
a.link:hover {  
  background-color: purple;  
}
```

transition-property

Określa nam, na które właściwości CSS wpłynie funkcja przejścia.

Wartość **“all”** odpowiada za wszystkie właściwości.

Jeśli chcielibyśmy animować tylko kolor tekstu, to wpisalibyśmy `color`, jeśli tło to `background-color` itd.



```
.property {  
  transition-property: color;  
  transition-property: background-color;  
  transition-property: border;  
}
```

transition-duration

Czyli czas trwania przejścia.

Im dłuższy, tym bardziej subtelna będzie animacja.

Pamiętajmy o dodawaniu jednostek
– sekund (s) lub milisekund (ms).



```
.property {  
  transition-duration: 0.7s;  
  transition-duration: 1s;  
  transition-duration: 5s;  
}
```

transition-timing-function

Czyli sposób, w jaki przejście będzie zachodzić.

Opisuje to funkcja czasu.

W CSS występuje kilka ich predefiniowanych rodzajów.

Funkcje można wyklikać na stronie:

<http://cubic-bezier.com>

[Codepen](#)

```
.property {  
  transition-timing-function: linear;  
  /* zmiana następuje wg funkcji liniowej */  
  transition-timing-function: ease;  
  /* zmiana następuje wg funkcji liniowej */  
  transition-timing-function: ease-in;  
  /* animacja na początku ma wolniejsze tempo,  
   i przyspiesza na końcu */  
  transition-timing-function: ease-out;  
  /* animacja na początku ma szybsze tempo,  
   i zwalnia na końcu */  
  transition-timing-function: ease-out;  
  /* animacja na początku ma szybsze tempo,  
   i zwalnia na końcu */  
  transition-timing-function: cubic-bezier();  
  /* przejście następuje wg zadanej funkcji Beziera */  
}
```



Właściwość:
transform

Właściwość: transform

Dzięki właściwości transform możemy przekształcać nasze obiekty w różny sposób. Zajmiemy się przypadkami w przestrzeni 2D.

Możliwe operacje to skalowanie (powiększanie, pomniejszanie), przesuwanie po osi X i po osi Y, obrót oraz pochylenie.

[Codepen](#)

```
.translate {  
    /* przesuwanie */  
    transform: translateX(150px);  
    transform: translateX(-150px);  
    transform: translateY(150px);  
    transform: translateY(-150px);  
  
    /* obracanie */  
    transform: rotate(45deg);  
  
    rotateX(45deg);  
    rotateY(45deg);  
    rotateZ(45deg);  
  
    /* skalowanie */  
    transform: scale(1.2);  
    transform: scale(4);  
}
```





Tworzenie animacji
@keyframes

@keyframes

Tworzenie animacji następuje przez regułę **@keyframes**.

changeColors to nazwa animacji, którą nadajemy sami. Następuje zawsze po deklaracji **@keyframes**.

W środku animacji, pomiędzy nawiasami klamrowymi, definiujemy, co będzie działo się w danych przedziałach czasowych animacji.



```
@keyframes changeColors {  
  0% { background-color: green; }  
  50% { background-color: #263479; }  
  100% { background-color: green; }  
}
```

[Codepen](#)

@keyframes

W przypadku CSS przypisuje się to postępowi animacji, wyrażonemu w procentach, gdzie **0%** to początek animacji, zaś **100%** to jej koniec.



```
@keyframes changeColors {  
  0% { background-color: green; }  
  50% { background-color: #263479; }  
  100% { background-color: green; }  
}
```

@keyframes

Dodaliśmy dwie nowe właściwości:
animation-name, której wartością jest nazwa naszej animacji (**changeColors**), oraz **animation-duration**, czyli dowolny czas trwania animacji.

Dzięki tym dwóm właściwościom dodaliśmy do naszego linku z klasą link stworzoną przez nas animację.



```
.animation {  
  background-color: green;  
  padding: 20px 100px;  
  margin: 50px 50px;  
  font-size: 60px;  
  /* deklaracja animacji */  
  animation-name: changeColors;  
  animation-duration: 5s;  
}
```

@keyframes - właściwości



```
.animation {  
  animation-name /* nazwa animacji */  
  animation-duration /* czas trwania animacji */  
  animation-timing-function /* funkcja czasowa animacji;  
    mamy do dyspozycji te same funkcje, co w przypadku transition */  
  animation-delay /* opóźnienie, czyli czas po jakim animacja zacznie działać */  
  animation-direction /* kierunek rozpoczęcia i powtarzania się animacji,  
    więcej o tej właściwości poniżej */  
  animation-iteration-count /* liczba powtórzeń (iteracji) animacji */  
  animation-fill-mode /* określa jakie cechy będzie miał animowany obiekt,  
    gdy jego animacja będzie gdy animacja się zakończy i nie będzie powtarzana. */  
  animation-play-state /* zatrzymywanie bądź wznowienie animacji,  
    dozwolone wartości to paused (zatrzymana) i running (w toku) */  
}
```



Homework

Zadanie domowe 5

Dodać efekty i animacje na elementach strony:

- Efekt hover na **linkach** i **buttonach**
- **Powiększenie się zdjęcia w galerii** po najechnaniu myszką
- **Pojawianie się nazwisk członków zespołu** po najechnaniu na ich twarze oraz **podświetlenie ich**



Linki pomocnicze

- Wszystkie informacje znajdziecie na naszym [GitHubie](#)
- Pytania prosimy zadawać w formie Issue lub na adres frontend@daftcode.pl
- Miejsce inspiracji [CodePen.io](#)
- Wszystkie przykłady z prezentacji znajdziesz [tutaj](#)





Pytania?