

# TRANSITION I TRANSFORM

Dwa najbliższe dni.

Projekty przede wszystkim.

Zobaczymy też jQuery w działaniu.

# ANIMACJE

Ożywianie strony za pomocą CSS.

Nie da się zrobić nie tylko atrakcyjnej wizualnie, ale i prostej w obsłudze strony bez animacji. Animacje są kluczowym aspektem w pracach nad UX strony, pomagają też stworzyć intuicyjny interfejs co zwiększa konwersję.

# Transition (przejście)

Podstawowa animacja w CSS.

Używana często po najechaniu, ale też z użyciem JS/JQ przy zmianie klasy.

# Transition

Zmiana dokona się domyślnie od razu po najechaniu na element

```
button {  
    background-color:white;  
    color: black;  
}
```

```
button:hover {  
    background-color: black;  
    color: white;  
}
```



# Transition

W istocie mamy coś takiego.  
Czyli zmiana odbywa się od  
razu (zmiana trwa 0 sekund).

```
button {  
    background-color:white;  
    color: black;  
    transition: 0s;  
}
```

```
button:hover {  
    background-color: black;  
    color: white;  
}
```

# Transition

Ale zmianę możemy wydłużyć w czasie. i sprawić, że przeglądarka będzie ją animować.

```
button {  
    background-color:white;  
    color: black;  
    transition: 1s;  
}
```

```
button:hover {  
    background-color: black;  
    color: white;  
}
```

# Transition

Zwróćmy uwagę, że zmianę ustawiamy na elemencie głównym a nie na triggerze (wyzwalaczu).

```
button {  
    background-color:white;  
    color: black;  
    transition: 1s;  
}
```

```
button:hover {  
    background-color: black;  
    color: white;  
}
```

# Transition

Do przejścia (transition) dochodzi tylko wtedy gdy zmianie ulegają jakieś właściwości.

```
button {  
    background-color:white;  
    color: black;  
    transition: 1s;  
}
```

```
button:hover {  
    background-color: black;  
    color: white;  
}
```



# Transition

```
button {  
    background-color:white;  
    color: black;  
    transition: 1s linear;  
}
```

```
button:hover {  
    background-color: black;  
    color: white;  
}
```

Przejście wymaga:

1. zadeklarowania wartości początkowej.
2. zadeklarowania wartości końcowej.
3. określenie samego przejści (właściwość transition).
4. zdefiniowany mechanizm uruchomienia (np. :hover).

# Transition

```
button {  
    background-color:white;  
    color: black;  
    transition: 1s linear;  
}
```

```
button: hover {  
    background-color: black;  
    color: white;  
}
```

Zdefiniowany mechanizm uruchomienia.

Ten mechanizmu odpowiada też za cofnięcie zmiany w ten sam sposób jaki zaistniała. Czyli jeśli kursor myszki przesunie się poza obszar przycisku albo zabierzemy klasę to następuje powrót do właściwości wejściowej (również za pomocą przejścia).

# Transition

Zdefiniowany mechanizm uruchomienia.

```
button {  
    background-color:white;  
    color: black;  
    transition: 1s;  
}
```

```
buton.show {  
    background-color: black;  
    color: white;  
}
```

Najczęściej spotykany sposób to dodanie klasy (np. w JS czy JQ).

# Transition - właściwości podlegające przejściu

transition: 1s;

transition: all 1s;

Jeśli nie podamy właściwości, które mają ulegać przejściu to domyślnie ustawiona jest wartość all (czyli wszystkie, które się zmieniają).

transition: all 0s ease; /\* Domyślnie \*/

# Transition - właściwości podlegające przejściu

**transition-property:** właściwość lub all

np.

**transition-property:** font-size;

w zapisie skróconym

**transition:** font-size 1s;

# Transition - czas trwania przejścia

**transition-duration**: w sekundach lub milisekundach

np.

**transition-duration: .5s; //pół sekundy**

w zapisie skróconym

**transition: 500ms; transition 0.4s; transition: 2000s;**

Transition ma wartość, którą każdy element ma ustawioną na 0 sekund, czyli przejście jest natychmiastowe (czyli też bez animacji).

**transition: 0s; //domyślnie**

# Transition - funkcja czasu

`transition-timing-function: m.in. ease | linear | cubic-bezier`

w zapisie skróconym

`transition: all 0.5s ease;`

`ease` - zacznij wolno przyśpiesz na końcu zwolnij.

`linear` - ta sama prędkość podczas całej animacji.

`ease-in` - zacznij wolno i przyśpieszaj

`ease-out` - zacznij szybko i zwalniaj

`cubic-bezier(a,b,c,d)` - zobaczmy na przykładzie

`transition-timing-function: ease; //domyślnie`

# Transition - opóźnienie

transition-delay: 0.2s

w zapisie skróconym

transition: all 0.5s ease .2s;

transition: 0.5s .2s; //ważne by opóźnienie było jako druga wartość, bo pierwszą zawsze jest czas trwania animacji.

transition-delay: 0; //domyślnie od razu



# Transition - deklarowanie różnych przejść dla różnych właściwości.

Możemy określić więcej niż jedno przejście, tzn. określi przejścia dla różnych właściwości. Można to zadeklarować w zapisie skróconym.

**transition:** color 1s ease, font-size 1s 2s linear, top 1s ease-out;

Ps. opóźnienie tak samo przy powrocie.

# Transition - które właściwości warto animować?

**kolory:** background-color, color, opacity

**wielkości:** width, height,

**pozycja:** top, bottom, left, right

**elementy modelu pudełkowego:** padding, border, margin

**transformacje:** m.in. rotate, scale, translate

**inne:** cienie (text-shadow i box-shadow)

\* transformacji ulegają różne jednostki wielkości: piksele i procenty są ok. Obsługiwana jest też funkcja **calc()**

# Transition - które właściwości warto animować?

Nie podlegają przejściu takie właściwości jak **display**, **float**, **position**, **rodzaj czcionki**.