StillCo
IT & C SPECIALISTS
edu

ОСНОВИ ПРОГРАМУВАННЯ
ДЛЯ ШКОЛЯРІВ

ГРУПА
PRO

СТВОРЕННЯ САЙТУ
HTML ТА CSS

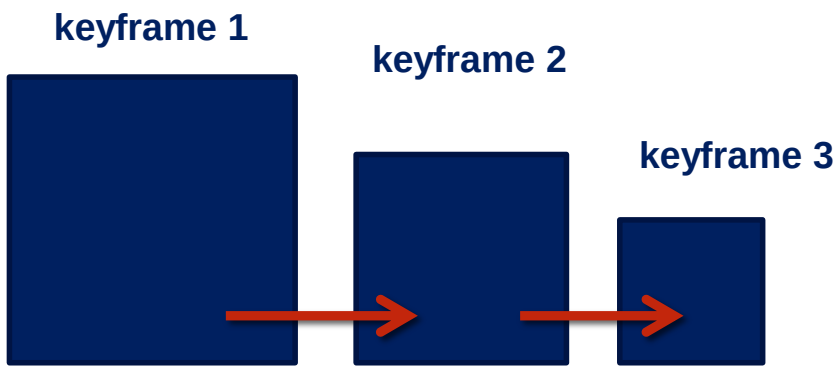
АНІМАЦІЯ

ЧАСТИНА 1

Ми часто бачимо на сайтах анімації, від простих змін розмірів вікон до якихось спливаючих текстів або картинок. Не завжди для цього потрібно застосування важких кодів Java Script або інших мов програмування. Краще створити гарну анімацію на CSS ніж виконати поганий код, який буде лагати. CSS-анімації можуть відтворюватися без додаткових дій з боку користувача і складатися з кількох кроків.

Спочатку, давайте розберемо, з чого складається будь-яка анімація. Будь яка анімація складається з ключових кадрів – **keyframe**. Як в мультику – один кадр має змінитися іншим. В програмуванні так само: ми створюємо декілька кадрів, та прописуємо яким чином вони мають змінювати один одного.

Як приклад: анімація де змінюється розмір квадрата: То тепер тільки треба задати яким чином все буде змінюватися.

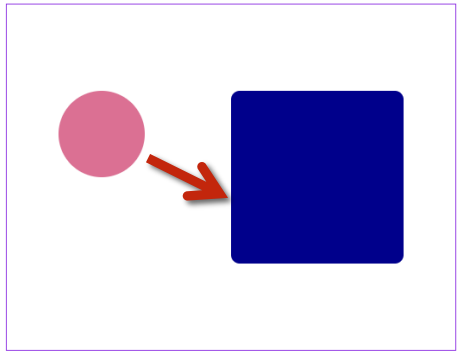



```
27  .block1  {
28    width: 100px;
29    height: 100px;
30  }
31  .block2  {
32    width: 70px;
33    height: 70px;
34  }
35  .block3  {
36    width: 50px;
37    height: 50px;
38  }
```

@keyframes

Щоб розповісти браузеру, з чого почати та чим закінчити анімацію, в CSS використовується директива **@keyframes.**

Уявимо, що у нас є два блоки: рожеве коло та синій квадрат. Ми хочемо написати анімацію так, щоб рожевий круг перетворювався на синій квадрат, а синій квадрат перетворювався на рожевий круг.
В цьому уроці ми зробимо першу дію, другу ви зробите самостійно.



Розпочати створення нашої анімації треба з розкладання її на кроки - ключові кадри. Наша анімація, на початку, буде проста, у неї буде всього два ключових кадри.

Щоб перетворити рожеве коло на синій квадрат, нам потрібно буде поміняти чотири властивості:
width,
height ,
background-color
border-radius

```
35  .circle  {
36    width: 100px;
37    height: 100px;
38    background-color: palevioletred;
39    border-radius: 50%;
40  }
41  .patrat  {
42    width: 200px;
43    height: 200px;
44    border-radius: 10px;
45    background-color: darkblue;
46  }
```

Щоб прописати ключові кадри, використовуємо директиву **@keyframes:** Після ключового слова **@keyframes** ми маємо написати ім'я анімації. Воно знадобиться нам, щоб зв'язати анімацію для конкретного елемента із ключовими кадрами. Бажано, щоб ім'я анімації було унікальним. В нашому випадку це **circle-to-patrat**

```
@keyframes circle-to-patrat {
  from {
    width: 100px;
    height: 100px;
    background-color: palevioletred;
  }
  to {
    width: 200px;
    height: 200px;
    background-color: darkblue;
  }
}
```

Якщо в коді зустрічається кілька директив з однаковими іменами, то відтворюватиметься остання анімація, що стоїть нижче в коді.

Ключові кадри можуть прописуватися за допомогою ключових слів **from** (початковий кадр) та **to** (кінцевий кадр). Це зручно, якщо у вас всього два ключові кадри. Якщо кадрів більше двох, то можна використовувати відсотки.
Браузер розшифровує ключове слово **from** як **0%,** а ключове слово **to** як **100%.** Наприклад, додамо ще один keyframe з проміжним кроком в **50%,** коли наше коло буде фіолетовим прямокутником.

```
@keyframes circle-to-patrat {
  from {
    width: 100px;
    height: 100px;
    background-color: palevioletred;
  }
  50% {
    width: 50px;
    height: 200px;
    background-color: #7F6EDB;
    border-radius: 50px;
  }
  to {
    width: 200px;
    height: 200px;
    background-color: darkblue;
    border-radius: 10px;

  }
}
```

Ми прописали ключові кадри анімації, але поки що нічого не відбувається

Щоб анімація почала програватись, нам потрібно прописати анімацію якомусь елементу, щоб браузер розумів, який елемент на сторінці анімувати та час.

animation-name

Для присвоєння анімації елементу саме потрібне ім'я анімації, яке ми вигадали.

```
.child-one {
  animation-name: circle-to-patrat;
}
```

```
68 .circle {
69   animation-name: circle-to-patrat;
70   animation-duration: 5s;
71 }
72
```

animation-duration

За допомогою якості **animation-duration** пропишемо тривалість одного циклу анімації. Значення цієї властивості вказується у секундах **s** або мілісекундах **ms.**

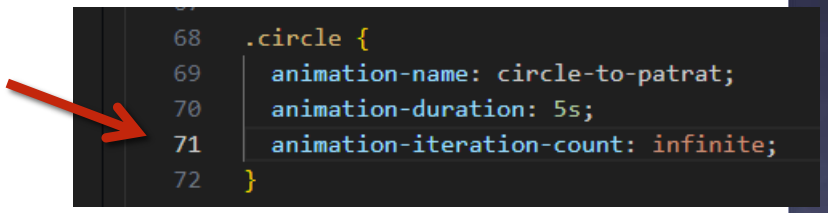
Нехай коло перетворюється на квадрат за 5 секунд.

Если указать 0s, то ключевые кадры будут пропущены, анимация применится мгновенно.

animation-iteration-count

За допомогою якості **animation-iteration-count** можна вказати, скільки разів анімація буде програватись.

Як значення вказується число, що означає кількість повторень або ключове слово **infinite**. Якщо зазначено **infinite**, то анімація повторюватиметься нескінченно. Це значення зустрічається найчастіше!



Тепер анімація програється постійно, але ви, напевно, бачите, що після останнього кадру відбувається різкий стрибок до початкового стану. Тобто, одразу після третього кадру перестрибує на перший без проміжного.

animation-direction

Властивість **animation-direction** повідомляє браузеру, чи анімація повинна програватися у зворотному порядку.

Доступні значення:

normal — значення за замовчанням, анімація відтворюється від початку до кінця, після чого повертається до початкового кадру.

reverse - анімація програється у зворотному порядку, від останнього ключового кадру до першого, після чого повертається до останнього кадру.

alternate - кожен непарний повтор (перший, третій, п'ятий) анімації відтворюється у прямому порядку, а кожен парний повтор (другий, четвертий, шостий) анімації відтворюється у зворотному порядку.

alternate-reverse - аналогічно значенню alternate, але парні та непарні повтори змінюються місцями.

Нам підійде значення
animation-direction: alternate;

Після чого анімація красиво програється. Коло плавно стає квадратом, а потім знову плавно перетворюється на коло

```
68  .circle {
69    animation-name: circle-to-patrat;
70    animation-duration: 5s;
71    animation-iteration-count: infinite;
72    animation-direction: alternate;
73  }
74
```

Давайте зробимо ще один об'єкт – квадрат , який буде перетворюватися на круг. Для того, щоб не робити знову окремі **@keyframe** для квадрата, ми можемо йому задати ту саму назву анімації, тільки зробити **animation-direction: alternate-reverse;**
Тобто його перетворення буде відбуватися в зворотному напрямку.

```
69  .circle {
70    animation-name: circle-to-patrat;
71    animation-duration: 5s;
72    animation-iteration-count: infinite;
73    animation-direction: alternate;
74  }
75
76  .patrat {
77    animation-name: circle-to-patrat;
78    animation-duration: 5s;
79    animation-iteration-count: infinite;
80    animation-direction: alternate-reverse;
81  }
```