ПРАКТИЧНА РОБОТА №5

Тема: Анімація на CSS3.

Мета: Практично познайомитися із принципами та синтаксисом анімації на вебсторінці з використанням CSS3.

Знати: Основні команди, за допомогою яких можна на CSS3 анімацію.

Вміти: Компонувати текстову або графічну інформацію для організації анімаційних ефектів.

Теоретичні відомості

CSS анімації роблять можливим анімацію переходів (transitions) з однієї конфігурації CSS стилю до іншої. Анімації складаються з двох компонентів, а власне - стилю, котрий описує CSS анімацію та набір ключових кадрів (keyframes), які задають початковий та кінцевий стан стилю анімації, а також є можливим задання точок проміжного стану.

В CSS анімацій є три ключові переваги перед традиційними скриптовими техніками анімації:

- Вони легкі у використанні для простих анімацій; ви можете створити їх навіть без знань JavaScript.
- Анімації добре функціонують, навіть при помірному навантаженні на систему. Прості анімації на JavaScript можуть працювати не дуже добре (в разі, якщо вони не ідеально зроблені). Рушій рендера може використовувати техніку пропуску кадрів та інші засоби для підтримки гладкої анімації.
- Надаючи таким чином браузеру контроль над послідовністю анімації ви надаєте можливість браузеру оптимізувати свою роботу та ефективність, наприклад, завдяки зупинці анімації у вкладках, які не є відкриті для перегляду.

Синтаксис анімації

Щоб скористатися анімацією в своєму проекті, вам досить зробити дві речі:

1. Створити саму анімацію.
2. Підключити її до елементу, який треба анімувати, і вказати необхідні якості.

Анімація являє собою набір ключових кадрів, який міститься в css і виглядає ось так:

```
@keyframes назва_анімації {
    0% {
        стан змінних параметрів
    }
    ....
    100% {
        стан змінних параметрів
    }
}
```

Анімацію необхідно підключити до елемента, який будемо анімувати. Для цього до об’єкту додаємо характеристику **animation**:

```
Селектор {animation: назва_анімації час_анімації зацикленість
модель_анімації;}
```

Ключові кадри

Ключові кадри використовуються для вказування значень властивостей анімації в різних точках анімації. Ключові кадри визначають поведінку одного циклу анімації; анімація може повторюватися нуль або більше разів. Ключові кадри вказуються за допомогою правила **@keyframes**

Створення анімації починається з установки ключових кадрів правила **@keyframes**. Кадри визначають, які властивості на якому етапі будуть анімовані. Кожен кадр може включати один або більше блоків оголошення з одного або більше пар властивостей і значень. Правило **@keyframes** містить ім'я анімації елемента, яке пов'язує правило і блок оголошення елемента.

```
@keyframes shadow {
   from {text-shadow: 0 0 3px black;}
   50% {text-shadow: 0 0 30px black;}
   to {text-shadow: 0 0 3px black;}
}
```

Ключові кадри створюються за допомогою ключових слів from і **to** (еквівалентні значенням **0%** і **100%**) або за допомогою процентних пунктів, яких можна задавати скільки завгодно. Також можна комбінувати ключові слова і процентні пункти. Якщо кадри мають однакові властивості і значення, їх можна об'єднати в одне оголошення:

```
@keyframes move {
   from,
   to {
      top: 0;
      left: 0;
```

```
    }
    25%,
    75% {top: 100%;}
    50% {top: 50%;}
}
```

Якщо 0% або 100% кадри не вказані, то браузер користувача створює їх, використовуючи обчислювані (спочатку задані) значення анімованої властивості.

Якщо кілька правил @keyframes визначені з одним і тим же ім'ям, то спрацює останнє в порядку документа, а всі попередні проігноруються.

Після оголошення правила @keyframes, ми можемо посилатися на нього у властивості animation:

```
h1 {
    font-size: 3.5em;
    color: darkmagenta;
    animation: shadow  2s infinite ease-in-out;
}
```

Конфігурування анімації

Щоб створити CSS-анімаційну послідовність, ви стилізуєте анімований елемент **animation** властивістю чи її підвластивостями. Це дозволяє вам коригувати таймінг, тривалість та інші деталі анімації згідно з вашими потребами. Це конфігурує актуальний вигляд анімації, яка здійснюється проходом через @keyframes правила,.

Підвластивостями властивості animation є:

Таблиця 5.1 - Властивості анімації в CSS

animation-delay	Змінює час затримки між часом з моменту завантаження елемента та початком анімаційної послідовності.
animation-direction	Визначає зміну напрямку анімації та його чергування в залежності від кількості проходів анімації, а також може задавати повернення в початковий стан і починати прохід заново.
animation-duration	Визначає тривалість циклу анімації.
animation-iteration-count	Визначає кількість проходів (повторів) анімації; ви можете також обрати значення infinite для нескінченного повтору анімації.
animation-name	Задає ім'я для анімації @keyframes через at-правило, яке описує анімаційні ключові кадри.
animation-play-state	Дозволяє вам призупиняти та відновлювати анімацію.
animation-timing-function	Задає конфігурацію таймінгу анімації; інакше кажучи, як саме буде анімація робити прохід через ключові кадри, це можливо

	завдяки кривим прискорення.
animation-fill-mode	Визначає, які значення будуть застосовані для анімації перед початком та після її закінчення.

Всі ці правила анімації можна застосовувати окремо. Проте, можна використати скорочені правила. **animation** скорочення є вигідним для економії місця. От, для прикладу, правила анімації, які ми вже використали в статті:

```
p {
  animation-duration: 3s;
  animation-name: slidein;
  animation-iteration-count: infinite;
  animation-direction: alternate;
}
```

Їх можна замінити ось таким чином

```
p {
  animation: 3s infinite alternate slideIn;
}
```

Приклад анімації

Приклад реальної анімації спробуємо показати на реалізації орбіти Місяця довкола Землі. Для цього створюємо початковий HTML-файл.

```
<!DOCTYPE html>
<html lang="uk">
  <head>
    <meta charset="UTF-8">
    <title>Орбіта місяця на CSS3</title>
  </head>
  <body>

  </body>
</html>
```

Додаємо до нього картинки Землі та Місяця (взяти в долучених файлах завдання). Картинкам присвоюємо відповідні ідентифікатори, через які будемо до них звертатися.

```
    <img src="images/earth.png" id="earth">
    <img src="images/moon.png" id="moon">
```

Далі в блок HEAD додаємо блок STYLE. І решту налаштувань будемо робити в цьому блоці.

Початкову картинку отримуємо в такому виді

Рисунок 5.1 - Початкове розташування об'єктів

Для того, щоб відразу прибрати всі зайві рамки в об’єктах, додамо код

```
body{
        margin: 0;
        padding: 0;
    }
```

Всі параметри стану об’єктів будемо додавати в

```
    #earth {
          /*Тут будуть параметри Землі */
    }
    #moon {
          /*Тут будуть параметри Місяця */
    }
```

Здвинемо Землю від краю вікна, щоб було місце для руху Місяця

```
        padding: 100px;
```

Для того, щоб Місяць міг рухатися в площині Землі, його потрібно вивести із загального потоку документа та вказати координати розташування приблизно в центрі. Нагадаємо, що координата 0:0 знаходиться в лівому верхньому куті вікна.

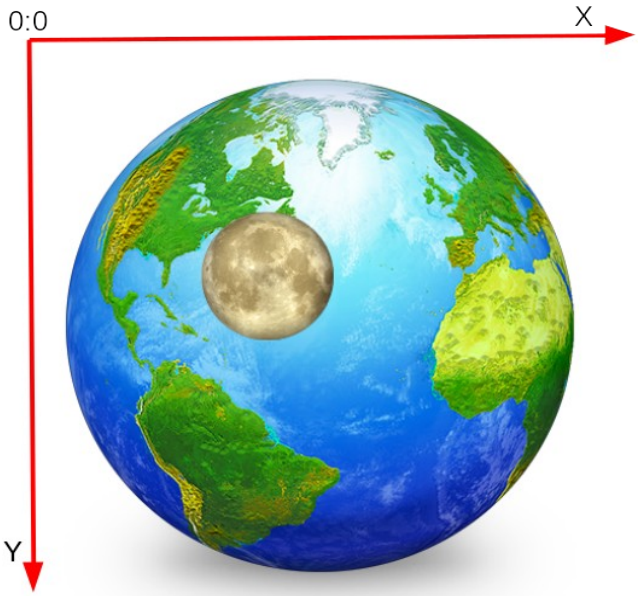

Рисунок 5.2 - Координати розташування на екрані

Далі координати можна вирахувати з математичною точністю або підібрати методом спроб.

```
position: absolute;
top: 250px;
left: 250px;
```

Отримуємо таку картинку

Рисунок 5.3 - Розташування Місяця над Землею

Для того, щоб почати робити анімацію додаємо до стилів перші ключові кадри

```
@keyframes orbita {
```

```
        0% {}
        25% {}
        50% {}
        75% {}
        100% {}
    }
```

В межах ключових кадрів прописуються зміни стану об’єкту.

Оскільки необхідно, щоб Місяць рухався довкола Землі по певній орбіті, визначимо цю орбіту%

0% - Місяць повинен бути за Землею

25% - Місяць в правому верхньому куті

50% - Місяць перед Землею

75% - Місяць в лівому нижньому куті

100% - Співпадає з початковою позицією

Графічний приклад представлено на рисунку нижче.

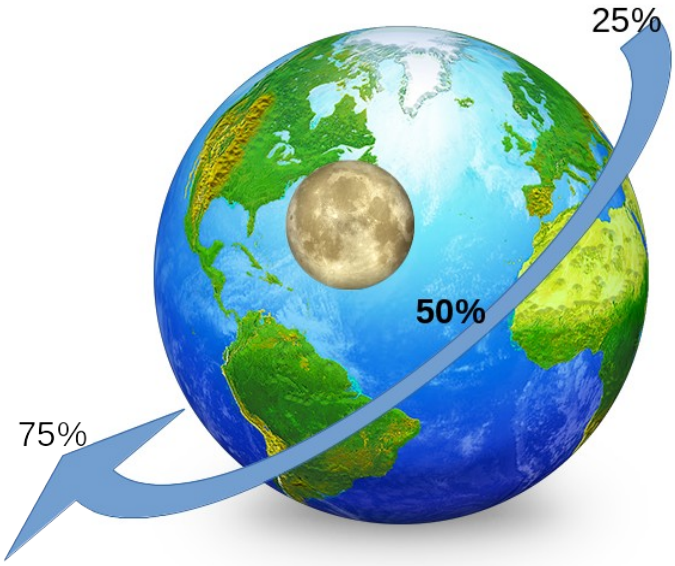

Рисунок 5.4 - Кадрові точки в руху Місяця

Проставимо координати розташування Місяця в цих точках:

```
@keyframes orbita {
        0% {}
        25% {top: 150px; left: 550px; }
        50% {top: 280px; left: 280px; }
        75% {top: 500px; left: 20px; }
        100% {}
    }
```

Тепер, щоб застосувати створені кадри необхідно додати до стилю Місяця параметри анімації

```
animation: 5s orbita  linear infinite;
```

де, 5s — час анімації

orbita — назва анімації (те як ми назвали перелік кадрів)

linear — спосіб анімації (конкретно тут лінійна функція, анімація відбувається рівномірно протягом усього часу, без коливань в швидкості)

infinite — зациклення анімації.

Перевіривши анімацію на цьому етапі, ви побачите, що Місяць рухається майже так як нам потрібно (як що це не так, то підберіть правильні координати у вашому прикладі). Проте не заходить за Землю, а рухається весь час перед нею. Щоб поправити це необхідно включити третю координату Z та підняти Рівень Землі на 1. А Місяць, проходячи за Землею буде рухатися в координаті z=0, а перед Землею — в координаті z=2.

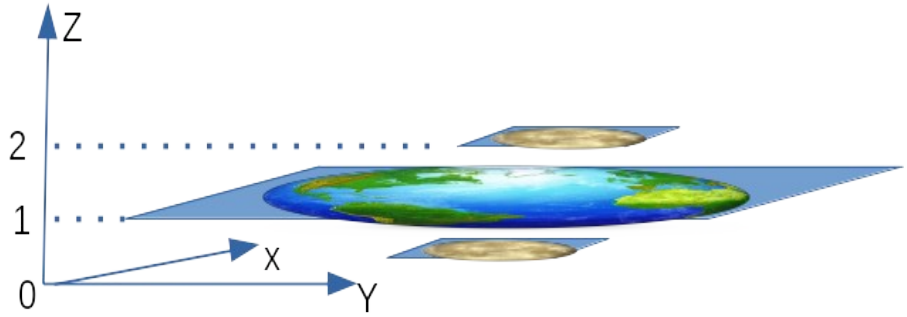

Рисунок 5.5 - Додаємо координату Z

Для Землі додамо код:

```
position: relative;
z-index: 1;
```

де, position: relative — виводимо елемент із загального потоку документу у відносне розташування,

z-index: 1 — змінюємо положення по координаті Z.

А для Місяця також додамо його початкову позицію по координаті Z:

```
z-index: 0;
```

А потім в кадрах анімації потрібно додати на 25% перехід місяця по координаті Z до значення 2, а на 75% перехід місяця по координаті Z до значення знову 0.

Отриманий результат потрібно перевірити і поправити у вашому конкретному випадку вказані значення.

Для того, щоб зробити ефект наближення Місяця в найближчій до нас точці і віддалення в найдальшій точці, необхідно збільшити ширину картинки Місяця в кадрі 50% та зменшити в кадрах 0% і 100%. Для додаткової плавності можна в проміжних кадрах додати проміжні значення. Наприклад,

```
0% {width: 110px;}
25% {width: 125px;}
50% {width: 140px;}
75% {width: 125px; }
100% {width: 110px;}
```

Тепер анімація стає схожою на задуману.

Для ще більшого ефекту можна додати обертання Місяця довкола власної осі (додайте до кадру 100%):

```
transform: rotate(360deg);
```

І на останок тло документа зробимо з картинкою космосу

```
body{
    background-image: url(images/cosmos.jpg);
    background-size: cover;
}
```

де, background-image: url(...) - картинка тла,

background-size: cover — розтягувати картинку до максимального розміру ширини або висоти.

Завдання

Відповідно до наведеного вище опису реалізувати анімацію орбіти Місяця довкола Землі.

Зверніть увагу, що в описі не подано повний код для того, щоб ви правильно дописали все необхідне самостійно.

Контрольні питання:

1. Які існують способи змінити стан об’єкту у вебдокументі?

2. Яким способом можна плавно змінити стан об’єкту при наведенні на нього показника миші?

3. Які варіанти трансформації об’єкту ви можете назвати?

4. В яких одиницях вимірюється і записується такий спосіб трансформації об’єкту як обертання?

5. Для чого використовується в CSS3 команда @keyframes?

6. Що таке кадри анімації в CSS3 та як їх використовувати?

7. Яким способом можна підключити анімацію до об’єкту у вебдокументі?

8. Що таке спосіб анімації та які способи анімації об’єктів можна підключати?

9. Як працює Z-координата у позиціюванні об’єктів у вебдокументі.

10. У яких випадках зміна Z-координати може не спрацьовувати?