**Лабораторная работа № 7**

**Преобразования. Переходы. Анимация**

**Цель работы:** изучить свойства преобразования, применение анимации к элементам, научиться использовать переходы.

**Теоретические сведения для выполнения работы**

**Преобразования**

К свойствам преобразованиям объекта относится вращение, масштабирование, смещение и наклон. Преобразование осуществляется с использованием свойства *transform*. Свойство имеет следующие значения:

1. **scale(0.5)** — увеличить или уменьшить в размерах элемент (например, для значения 0.5 в два раза);

2. **rotate(45deg)** — повернуть элемент на определенный угол заданный в deg;

3. **translate(40px, 60px)** — смещение элемента из его текущей позиции на некоторое расстояние вправо на 40px и вниз на 60px;

4. **skew(15deg)** — изменяет форму элемента, смещая верхний край элемента в одну сторону, а нижний — в противоположную.

По умолчанию точкой трансформации является центр элемента, но ее можно изменить с помощью свойства *transform-origin*.

Вращения и перемещения могут производиться во всех трех измерениях — X, Y и Z. Значение *rotateZ* эквивалентно *rotate*, так как выполняет вращение вокруг оси Z. Значение *rotateX* и *rotateY* вращают элемент вокруг горизонтальной оси X (наклоняя элемент вперед или назад) и вертикальной оси Y (поворачивая элемент вправо и влево) соответственно.

Также можно задать значение перспективы используя трансформацию **perspective()** или же свойство *perspective*. В качестве значения указывается расстояние от плоскости монитора до точки сходимости линий. Чем меньше значение, тем более выраженной выглядит перспектива. По умолчанию перспектива отображается так, как если бы наблюдатель находился прямо по центру элемента. Свойство *perspective-origin* смещает вправо или влево, вверх или вниз отображение перспективы.

Свойство *transform-style* позволяет располагать элементы в трехмерном пространстве, задавая значение **preserve-3d**.

**Переходы**

Переход представляет собой анимацию смены одного набора свойств CSS другим за определенный промежуток времени. Переходы задаются с помощью свойств *transition*. Свойство *transition-property* указывает, в каких свойствах должен присутствовать переход. Значение **all** означает, что переход задается для всех свойств, которые изменяются. Свойство *transition-duration* указывает продолжительность перехода к конечному значению, например, 0.5s.

Для замедления или ускорения перехода используется свойство *transition-timing-function* со значениями **linear**, **ease-in** и **ease-out**:

1. **linear** — переход изменяется с постоянной скоростью;

2. **ease-in** — изменение вначале протекает медленно, но ускоряется до самого конца перехода;

3. **ease-out** — изменение начинается быстро, но к концу перехода замедляется.

Также можно использовать значение **cubic-bezier**, которое определяет график изменения перехода по кривой Безье.

Для задержки перехода используется свойство *transition-delay* со значением, например, 0.5s. Для создания анимации при наведении мыши на элемент блока плавного перехода изменения цвета с зеленого на синий:

.div {

background-color: rgb(255,0,0);

transition: background-color 1s linear;

}

.div:hover {

background-color: rgb(0,0,255);

}

**Создание анимации**

Создание анимации проходит в два этапа.

1. Определение анимации. Включает настройку ключевых кадров со списком анимируемых CSS-свойств.

2. Применение анимации к элементу.

Кадры определяются с помощью правила @keyframes, структура которого имеет следующий вид:

@keyframes имяАнимации {

from {

/\* здесь перечисляются свойства CSS \*/

}

to {

/\* здесь перечисляются свойства CSS \*/

}

}

В блоке объявления селектора элемента, чтобы анимация к нему применялась используются следующая расширенная запись:

.anime {

animation-name: имяАнимации;

animation-duration: 2s;

animation-timing-function: ease-in-out;

animation-delay: 5s;

animation-iteration-count: 2;

animation-direction: alternate;

animation-fill-mode: forwards; }

Свойство *animation-name* задает имя анимации, определяемое правилом @keyframes, *animation-duration* задает длительность анимации, *animation-timing-function* задает временную функцию, описывающую ускорение и/или замедление воспроизведения анимации, *animation-iteration-count* задает количество повторений анимации, *animation-direction* и задает обратное воспроизведение анимации при повторном воспроизведении.

Чтобы анимация была бесконечной необходимо использовать значение infinite для свойства *animation-iteration-count.* Следует отметить, что для работы анимации необходимо задавать продолжительность и имя анимации.

Для того, чтобы использовать следующую запись, необходимо изначально для элемента задать свойство *position* со значением **absolute** или **relative**.

@keyframes around {

0% { left: 0; top: 0; }

25% { left: 200px; top: 0; }

50% { left: 200px; top: 200px; }

75% { left: 0; top: 200px; }

100% { left: 0; top: 0; }

}

p {

position: relative;

animation: around 4s linear infinite; }

**Задания к лабораторной работе № 7**

Создать новый HTML-документ с внешним подключением стилей, в котором должно быть следующие элементы:

**Задание 1** Создать четыре изображения, к которым необходимо добавить следующие преобразования:

**1.1** При наведении на первое изображение оно масштабируется в сторону уменьшения в 2 раза;

**1.2** Второе изображение повернуть на 45 градусов;

**1.3** Третье изображение сместить вправо на 50px;

**1.4** При наведении на четвертое изображение оно наклоняется на **–20** градусов.

**Задание 2** Расположите изображения согласно рис. 7.1 и расположите их следующим образом. Параметры перспективы выбрать произвольные

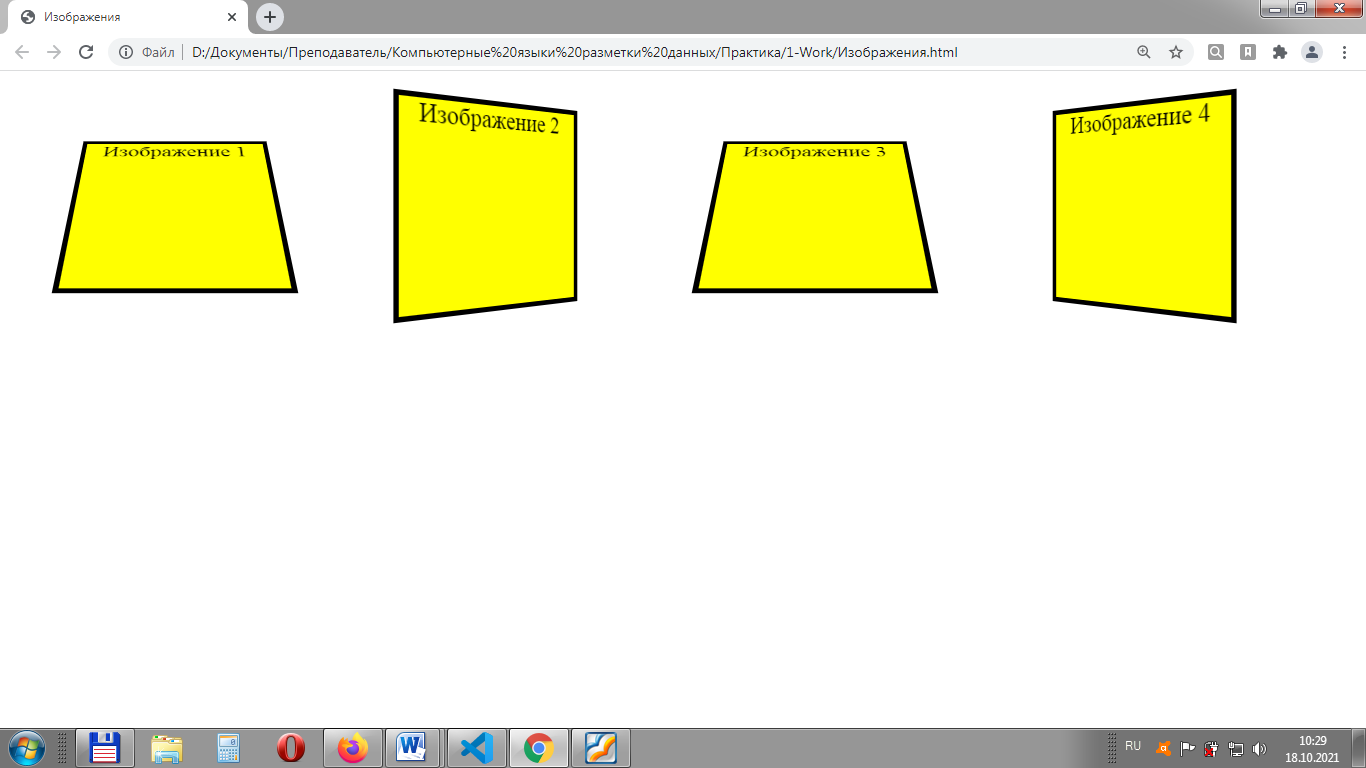


Рис. 7.1

**Задание 3** Создайте плавный переход цвета текста, выбрать произвольные цвета, время переход установить в 20 секунд, переход сделать линейным для блока размером 300px на 300px.

**Задание 4** Создайте анимацию непрерывной смены 5 изображений с продолжительностью 5 секунд с линейным ускорением. Изображения должны иметь общую границу произвольного цвета.

**Задание 5** Создайте из изображения 4 копии и для них примените следующие эффекты:

**4.1** Непрерывное перемещение 1 копии по горизонтали на 200px;

**4.2** Непрерывное перемещение 2 копии по вертикали на 200px;

**4.3** Непрерывное перемещение 3 копии по произвольной кривой;

**4.4** Непрерывное перемещение 4 копии по траектории описывающей квадрат, используя **left, top** для указания положения.

**Задание 6** Для каждого задания необходимо добавить заголовок, к которому применить непрерывное изменение прозрачности от 0 до 80% с обратным воспроизведением.

**Контрольные вопросы**

С помощью какого свойства осуществляет трансформация?

К свойствам преобразованиям объекта относится вращение, масштабирование, смещение и наклон. Преобразование осуществляется с использованием свойства *transform*. Свойство имеет следующие значения:

1. **scale(0.5)** — увеличить или уменьшить в размерах элемент (например, для значения 0.5 в два раза);

2. **rotate(45deg)** — повернуть элемент на определенный угол заданный в deg;

3. **translate(40px, 60px)** — смещение элемента из его текущей позиции на некоторое расстояние вправо на 40px и вниз на 60px;

4. **skew(15deg)** — изменяет форму элемента, смещая верхний край элемента в одну сторону, а нижний — в противоположную.

По умолчанию точкой трансформации является центр элемента, но ее можно изменить с помощью свойства *transform-origin*.

Вращения и перемещения могут производиться во всех трех измерениях — X, Y и Z. Значение *rotateZ* эквивалентно *rotate*, так как выполняет вращение вокруг оси Z. Значение *rotateX* и *rotateY* вращают элемент вокруг горизонтальной оси X (наклоняя элемент вперед или назад) и вертикальной оси Y (поворачивая элемент вправо и влево) соответственно.

Также можно задать значение перспективы используя трансформацию **perspective()** или же свойство *perspective*. В качестве значения указывается расстояние от плоскости монитора до точки сходимости линий. Чем меньше значение, тем более выраженной выглядит перспектива. По умолчанию перспектива отображается так, как если бы наблюдатель находился прямо по центру элемента. Свойство *perspective-origin* смещает вправо или влево, вверх или вниз отображение перспективы.

Свойство *transform-style* позволяет располагать элементы в трехмерном пространстве, задавая значение **preserve-3d**.

2. Как осуществить наклон?

**skew(15deg)** — изменяет форму элемента, смещая верхний край элемента в одну сторону, а нижний — в противоположную.

3. Как осуществить смещение?

**translate(40px, 60px)** — смещение элемента из его текущей позиции на некоторое расстояние вправо на 40px и вниз на 60px;

4. Как осуществляется вращение?

rotate

5. Каким образом масштабировать элементы?

**scale(0.5)** — увеличить или уменьшить в размерах элемент (например, для значения 0.5 в два раза);

6. Как создать перспективу изображения?

Также можно задать значение перспективы используя трансформацию **perspective()** или же свойство *perspective*. В качестве значения указывается расстояние от плоскости монитора до точки сходимости линий. Чем меньше значение, тем более выраженной выглядит перспектива. По умолчанию перспектива отображается так, как если бы наблюдатель находился прямо по центру элемента. Свойство *perspective-origin* смещает вправо или влево, вверх или вниз отображение перспективы.

Свойство *transform-style* позволяет располагать элементы в трехмерном пространстве, задавая значение **preserve-3d**.

7. Для чего используется свойство *transition*?

Переход представляет собой анимацию смены одного набора свойств CSS другим за определенный промежуток времени. Переходы задаются с помощью свойств *transition*. Свойство *transition-property* указывает, в каких свойствах должен присутствовать переход. Значение **all** означает, что переход задается для всех свойств, которые изменяются. Свойство *transition-duration* указывает продолжительность перехода к конечному значению, например, 0.5s.

Для замедления или ускорения перехода используется свойство *transition-timing-function* со значениями **linear**, **ease-in** и **ease-out**:

1. **linear** — переход изменяется с постоянной скоростью;

2. **ease-in** — изменение вначале протекает медленно, но ускоряется до самого конца перехода;

3. **ease-out** — изменение начинается быстро, но к концу перехода замедляется.

Также можно использовать значение **cubic-bezier**, которое определяет график изменения перехода по кривой Безье.

Для задержки перехода используется свойство *transition-delay* со значением, например, 0.5s.

8. Каким образом увеличить продолжительность перехода?

Transition-duration

8. Для чего используется *transition-timing-function*?

Для замедления или ускорения перехода используется свойство *transition-timing-function* со значениями **linear**, **ease-in** и **ease-out**:

1. **linear** — переход изменяется с постоянной скоростью;

2. **ease-in** — изменение вначале протекает медленно, но ускоряется до самого конца перехода;

3. **ease-out** — изменение начинается быстро, но к концу перехода замедляется.

Также можно использовать значение **cubic-bezier**, которое определяет график изменения перехода по кривой Безье.

9. Назовите этапы создания анимации

Свойство *animation-name* задает имя анимации, определяемое правилом @keyframes, *animation-duration* задает длительность анимации, *animation-timing-function* задает временную функцию, описывающую ускорение и/или замедление воспроизведения анимации, *animation-iteration-count* задает количество повторений анимации, *animation-direction* и задает обратное воспроизведение анимации при повторном воспроизведении.

Чтобы анимация была бесконечной необходимо использовать значение infinite для свойства *animation-iteration-count.* Следует отметить, что для работы анимации необходимо задавать продолжительность и имя анимации

10. Предназначение свойства *animation-direction*?

*animation-direction* и задает обратное воспроизведение анимации при повторном воспроизведении.

11. Каким образом сделать анимацию непрерывной?

Чтобы анимация была бесконечной необходимо использовать значение infinite для свойства *animation-iteration-count.*

12. Как изменить прозрачность элемента?

opacity

13. Каким образом создать несколько кадров анимации? Только 2 кадра?

14. Для чего необходимо свойство *animation-iteration-count*?

15. Что такое переходы?

Переход представляет собой анимацию смены одного набора свойств CSS другим за определенный промежуток времени.

16. Что будет происходить, если использовать *rotateX*?

* **rotateX()** — поворот элемента по оси X;

17. Что будет происходить, если использовать *rotateY*?

**rotateY()** — поворот по оси Y;