Концерн Беллегпром

УО «Барановичский государственный колледж легкой промышленности

имени В.Е. Чернышева»

Учебная дисциплина «Программные средства создания Internet-приложений»

**Инструкция**

по выполнению лабораторной работы №8

«Внедрение визуальных эффектов и анимации на HTML-странице»

Составитель: Самохвал Тамара Васильевна

Обсуждено и одобрено на заседании цикловой комиссии

дисциплин программного обеспечения

Протокол от \_\_\_\_\_\_\_\_\_\_\_\_ №\_\_\_\_\_\_\_\_\_\_

Барановичи

2020

**Цель: Сформировать умения по внедрению визуальных эффектов в HTML-документ**

**Оснащение: ПК, HTML-редактор, браузер**

**Ход работы**

1. **Изучите необходимые теоретические сведения и рассмотренные примеры, выполните задания по каждому из примеров:**

**1. Переходы**

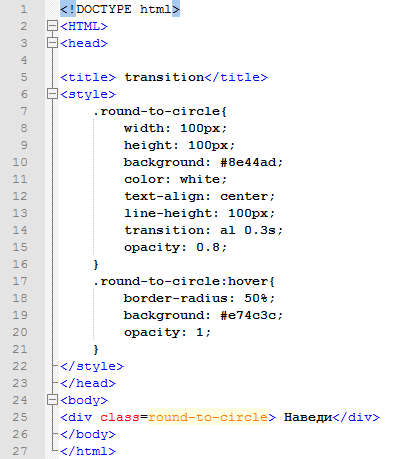
Переходом в css анимации называется плавная смена свойств элемента при каком-либо действии пользователя, чаще всего наведении на элемент курсора мыши. При использовании псевдокласса :hoverизменение происходит мгновенно, transitionже позволяет задать продолжительность и метод перехода.

**Синтаксис:** transition: [свойство,] длительность;

transition: all 1s; //для всех свойств длительность перехода 1с

transition: background 0.5s; //для фона длительность перехода 0,5с

**Пример.** При наведении на элемент div с классом round-to-circle, он из квадрата преобразуется в круг, у него изменяется цвет фона и непрозрачность становится равной 1. Все это происходит плавно, в течение 0.3с, т.к. задано свойство transition. Если это свойство не задано, то смена свойств будет происходить моментально.



|  |  |
| --- | --- |
| Исходное состояние | Состояние при наведении |
| Как создать css анимацию на css3 | Как создать css анимацию на css3 |

**Задание 1**

1. Реализуйте предложенный пример. Сохраните файл под именем **ЛР-7-perehod.html**

2. Измените значение в секундах до 1.5 секунд. Обновите страницу. Оцените разницу.

**2. Трансформация**

Трансформация – это изменение вида элемента, к которым относятся следующие визуальные модификации: поворот, масштабирование, наклон и сдвиг. Чтобы сделать трансформацию, к селектору добавляется свойство transform, а в качестве значения пишется функция трансформации, и её параметры. В общем виде это записывается так: transform: функция(параметры);

**2.1 Функция поворота – rotate()**

Поворачивает элемент на заданный угол

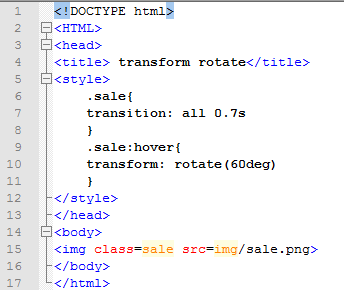
**Синтаксис:** transform: rotate(α)

Здесь: α – угол поворота. Положительное значение поворачивает элемент по часовой стрелке, отрицательное – против.

transform: rotate(90deg) //поворот по часовой на 90 градусов

transform: rotate(-180deg) //поворот против часовой на 180 градусов

**Пример.** При наведении на изображение, оно плавно поворачивается на 60 градусов по часовой стрелке.



|  |  |
| --- | --- |
| Исходное состояние | Состояние при наведении |
| http://web-grafika.pro/assets/images/web-grafika/animate_css/04_animate.jpg | http://web-grafika.pro/assets/images/web-grafika/animate_css/04_animate.jpg |

**Задание 2**

1. Реализуйте предложенный пример. Сохраните файл под именем **ЛР-7-rotate.html**

2. Изменяйте значение в секундах. Оцените разницу.

**2.2 Функция масштабирования – scale()**

Задаёт масштаб элемента по горизонтали и вертикали.

**Синтаксис:** transform: scale(sx[, sy]);

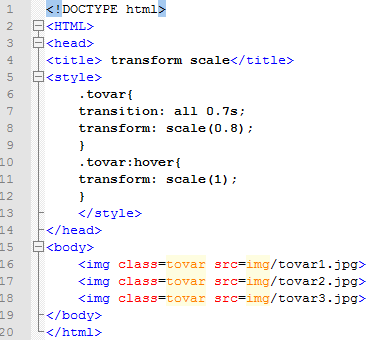
Здесь: sx – изменение масштаба по оси X; sy — изменение масштаба по оси Y. Значение больше 1 увеличивает масштаб элемента, меньше 1, наоборот, его уменьшает. Если задано только одно значение, то масштабирование будет происходить пропорционально в обе стороны: scale(1.2) соответствует scale(1.2, 1.2).

transform: scale(3, 1]); //увеличение ширины в 3 раза

transform: scale(1, 0.5); //уменьшение высоты в 2 раза

transform: scale(1.5); //увеличение размера в 1,5 раза

**Пример.** Уменьшим изображение до 80% от его первоначального состояния **–** scale(0.8). При наведении на изображение восстановим его прежнее состояние – scale(1).



|  |  |
| --- | --- |
| Исходное состояние | Состояние при наведении |
| http://web-grafika.pro/assets/images/web-grafika/animate_css/06_animate.jpg | http://web-grafika.pro/assets/images/web-grafika/animate_css/06_animate.jpg |

**Задание 3**

1. Реализуйте предложенный пример. Сохраните файл под именем **ЛР-7-scale.html**

2. Изменяйте значение в секундах и величину трансформации. Оцените разницу.

**2.3 Функция наклона – skew()**

Наклоняет элемент на заданное значение по горизонтали и вертикали в плоскости экрана.

**Синтаксис:** transform: skew(ax[, ay])

Здесь: ax – угол наклона по оси X; ay – угол наклона по оси Y. Если значение ay не указано, то оно считается равным 0: skew(30deg) соответствует skew(30deg, 0).

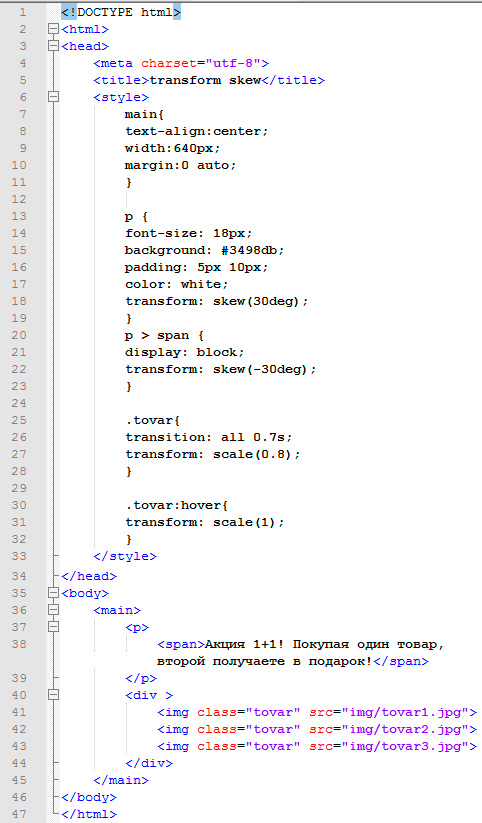
Положительное значение угла наклоняет влево, отрицательно значение – вправо

transform: skew(30deg, 60deg])

transform: skew(60deg, 30deg])

transform: skew(30deg) РАВНОСИЛЬНО transform: skew(30deg, 0)

**Пример.** В этом примере задали наклон 30 градусов для элемента p. Но нам нужен наклон только для фона параграфа, а сам текст должен быть выровнен прямо, поэтому положили текст внутри элемента span и задали ему компенсирующий наклон.



|  |  |
| --- | --- |
| До применения наклона | Наклон 30° для р |
|  |  |
| Наклон 30° для р и -30° для тега span, в котором находится текст | |
| http://web-grafika.pro/assets/images/web-grafika/animate_css/08_animate.jpg | |

**Задание 4**

Реализуйте предложенный пример. Сохраните под именем **ЛР-7-skew.html**

**2.4 Функция сдвига – translate()**

Сдвигает элемент на заданное значение по горизонтали и вертикали в плоскости экрана.

**Синтаксис:** transform: translate(tx[, ty])

Здесь: tx – значение смещения по оси X в пикселях, процентах или других единицах CSS; ty – смещение по оси Y. Если значение ty не указано, то оно считается равным 0: translate(30px) соответствует translate(30px, 0).

transform: translate(30px, 0]) РАВНОСИЛЬНО transform: translate(30px)

transform: translate(0, -30px]) //сдвиг вверх на 30px

transform: translate(60px, 30px])сдвиг вправо на 60px и вниз на 30px

transform: translate(-2%]) //сдвиг влево на 2%

**Задание 5**

1. Файл **ЛР-7-skew.html** сохраните под новым именем **ЛР-7-translate.html**

2. Кроме масштабирования добавьте перемещение вниз на 30px. Для этого измените стиль .tovar:hover так:

http://web-grafika.pro/assets/images/web-grafika/animate_css/09_animate.jpg

Теперь при наведении на изображение оно не только масштабируется, но и сдвигается вниз на 30px.



**3. CSS анимации**

Свойство animation позволяет создавать более сложные анимации. Эти анимации состоят из двух компонентов: стилевое описание анимации и набор ключей, определяющих начальное, конечное и, возможно, промежуточное состояние анимируемых стилей.

Создание css анимации начинается с установки ключевых кадров правила @keyframes. Кадры определяют, какие свойства, на каком шаге будут анимированы.

**Синтаксис:**

@keyframes название\_анимации {

from {свойство: параметры;}

50% {свойство: параметры;}

to {свойство: параметры;}

}

Ключевые кадры создаются с помощью ключевых слов fromи to(эквивалентны значениям 0% и 100%) или с помощью процентных пунктов, которых можно задавать сколько угодно.

После объявления правила @keyframes, необходимо ссылаться на него в селекторе в свойстве animation.

**Синтаксис, сокращенная запись:**

h1 {

animation: название\_анимации длительность [количество\_повторений];

}

**Обязательные параметры:**

* название анимации
* длительность

**Дополнительный параметр**

* количество повторений. Если не указано количество повторений – по умолчанию 1. Можно указывать любое другое число, либо использовать ключевое слово infinite – бесконечное количество повторений.

Таким образом, для задания css анимации элемента (селектора), полная запись будет выглядеть следующим образом:

h1 {

animation: название\_анимации длительность [количество\_повторений]; /\*в свойстве селектора h1 объявляем анимацию\*/

}

@keyframes название\_анимации { /\*описание анимации для селектора h1\*/

from {свойство: параметры;}

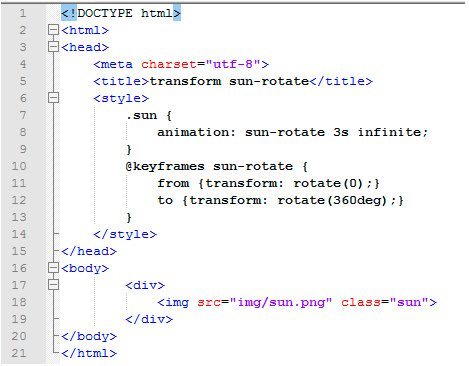
50% {свойство: параметры;}

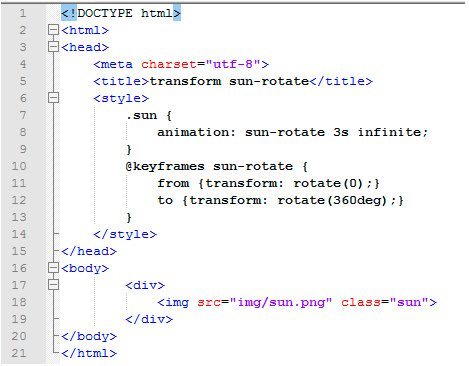
to {свойство: параметры;}

}

У css анимации есть и другие свойства. Например, задержка анимации, направление и т.д.

**Пример.** Применим к классу sun анимацию rotate длительностью 3 секунды с бесконечным количеством повторений.

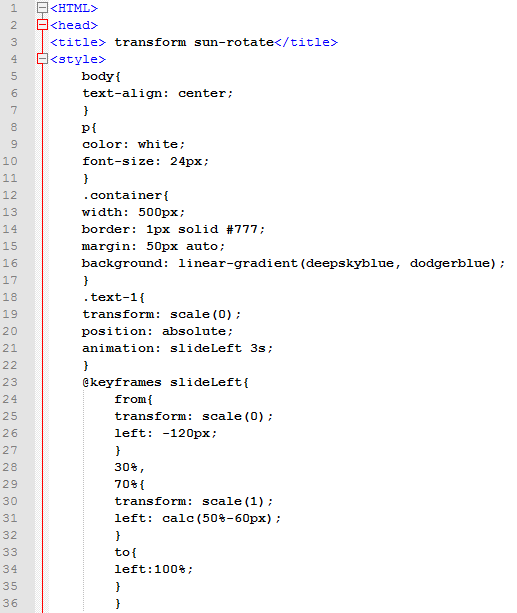


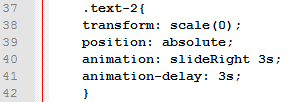


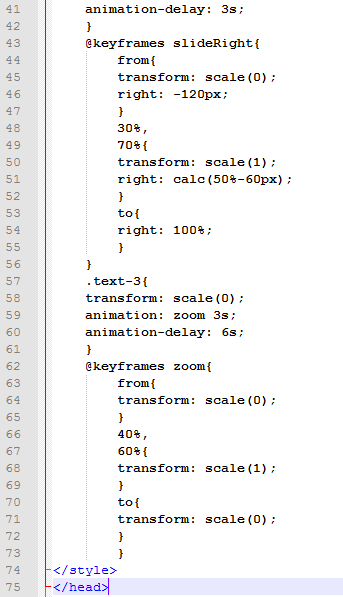
**Задание 6**

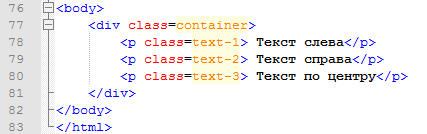
Реализуйте пример с крутящимся солнышком. Сохраните под именем **ЛР-7-sun.html**

**Пример.** Появление текста слева, справа и посередине.









**Задание 7**

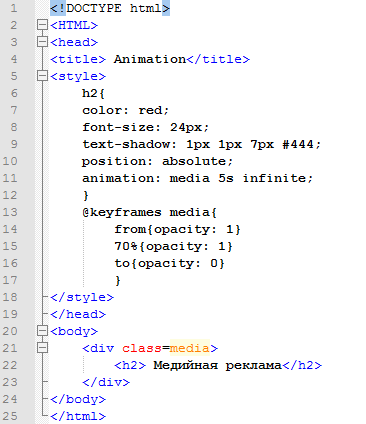
Реализуйте пример с движущимся текстом. Сохраните под именем **ЛР-7-text.html**

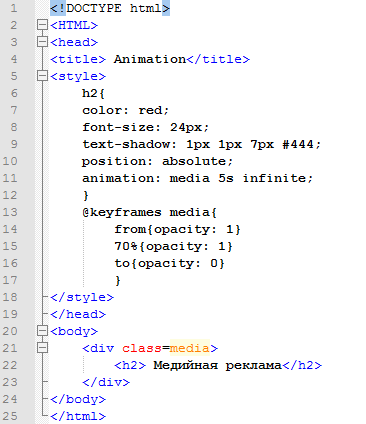
**Повторяющаяся анимация с плавной сменой итераций**

Плавный переход от итерации к итерации при повторяющейся анимации можно достичь, используя параметр прозрачности opacity.

**Задание 8**

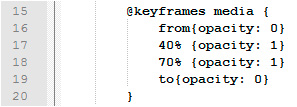
1. Реализуйте нижеприведенный код. Сохраните файл под именем **ЛР-7-media.html**





2. Просмотрите результат. Видно, что после плавного исчезновения текста он снова появляется, но достаточно резко. Это происходит потому, что в стилях в конце итерации непрозрачность равна нулю (строка 18 – to{opacity:0}), а в начале итерации непрозрачность равна 1 (строка 16 - from{opacity: 1}). Происходит резкая смена от полной непрозрачности к полной яркости.

3. Внесите изменения в эти свойства следующим образом – установите непрозрачность 0 в начале итерации (свойство from) и добавьте еще один временной промежуток 40% с непрозрачностью 1 и в результате появление текста в начале итерации также будет плавным.



**2. Выполните самостоятельные задания**:

1. Просмотрите css анимацию **itog.gif** в папке **itog**

2. Откройте в браузере файл **zadanie.html**. Как видите, пока анимация не работает.

3. Анимируйте шаблон **zadanie.html**. Для этого напишите стили для анимаций и примените их к соответствующим элементам в соответствии с файлом **itog.gif**:

* заголовок плавно появляется слева и останавливается примерно на середине фона;
* описание появляется справа и останавливается у левого края заголовка;
* иконки на фоне должны появляться после заголовка и описания;
* после фона с иконками плавно появляется кнопка;
* при наведении на кнопку, она плавно меняет фон.

4. Доработайте пункт 3 так, чтобы в конце всей анимации все элементы баннера кроме фона плавно исчезали, затем анимация должна начинаться сначала. Смена итераций должна быть плавной без резких скачков.

1. **Список контрольных вопросов**
   * + 1. Что такое css-переход? Как создается переход?
       2. Что такое псевдокласс :hover?
       3. Что такое css-трансформация? Назовите виды трансформации. Какими функциями они задаются?
       4. Из каких этапов (компонентов) состоит css-анимация? В чем они заключаются? Как реализуются?