JAVARUSH

Путь к карьере Frontend Fullstack разработчика

Модуль 1. WEB CORE

Уровень 12. Трансформации. Переходы и анимация. Часть 1.

JAVARUSH

CSS-трансформации

CSS-трансформации позволяют изменять положение, размер, поворот и наклон элементов без изменения их обычного потока документа. Это мощный инструмент для создания интерактивных и динамичных веб-интерфейсов.

Свойство transform позволяет применять двумерные и трехмерные трансформации к элементам. Оно объединяет различные функции трансформации:

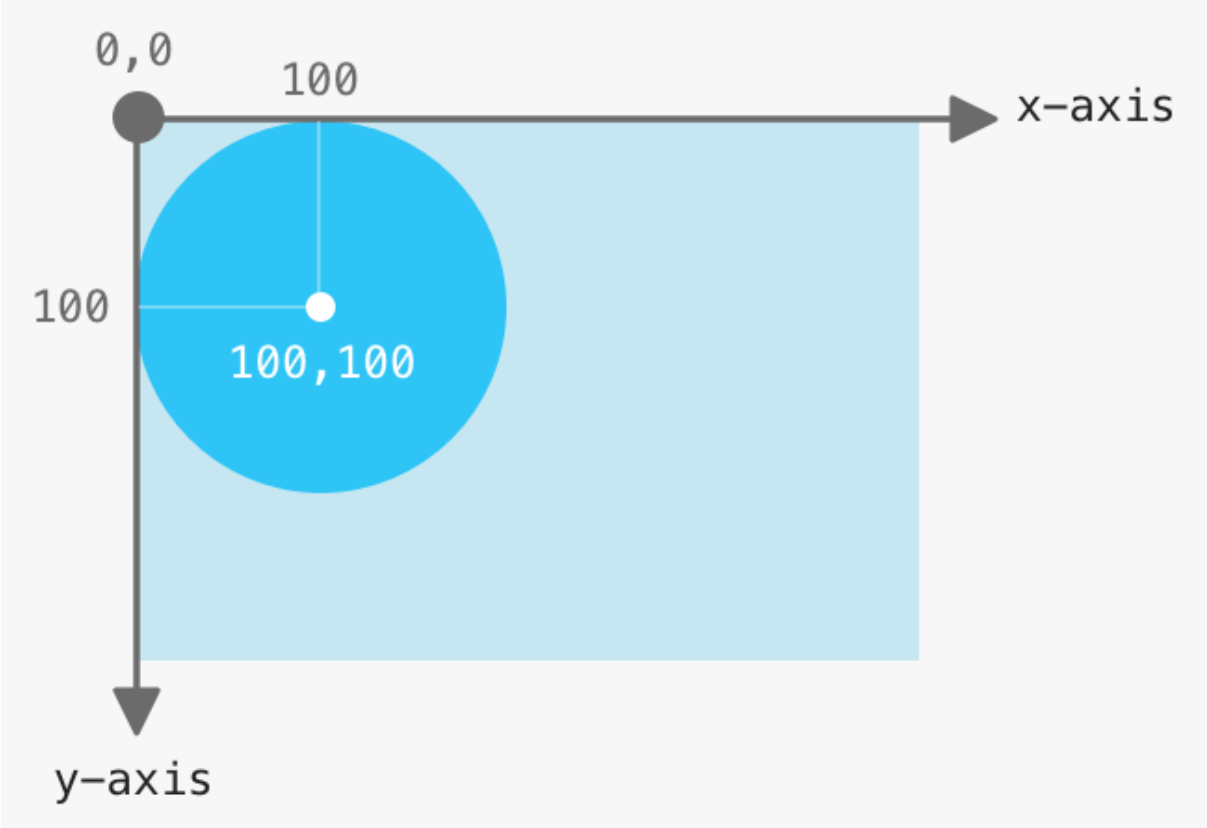
- translate() — перемещение объекта по координатам x и y
- rotate() — поворот объекта относительно его левого верхнего угла
- scale() — масштабирование объекта
- skew() — наклон элемента. В двумерном пространстве функция деформирует элемент

Пример использования:

```
.element {
   transform: translate(50px, 100px);
}
```

Система координат

В CSS система координат немного отличается от той, к которой вы привыкли в математике. Ось X по-прежнему направлена вправо, но ось Y инвертирована: положительные значения идут вниз, а отрицательные - вверх.



JAVARUSH

Точка отсчёта в CSS Transform

В CSS Transform, когда вы применяете преобразования к элементу (например, поворот или масштабирование), важно понимать, относительно какой точки эти преобразования происходят. По умолчанию, этой точкой является **верхний левый угол** элемента.

Это похоже на относительное позиционирование (position: relative), где вы можете смещать элемент относительно его исходного положения.

Даже если элемент визуально выглядит как круг, для CSS он всё равно остаётся прямоугольником. При применении трансформаций, таких как поворот, точка отсчёта будет находиться в левом верхнем углу этого прямоугольника.

Вы можете изменить точку отсчёта с помощью свойства transform-origin.

```
.circle {
   width: 100px;
   height: 100px;
   border-radius: 50%; /* Визуально делаем круг */
   outline: 1px solid red; /* Добавляем красную границу, чтобы увидеть реальные границы блока */
   transform: rotate(45deg); /* Поворачиваем элемент на 45 градусов */
}
```

JAVARUSH

Функции перемещения Transform: translate(X, Y)

Функция используется для смещения элемента вверх-вниз или влево-вправо. Она дает полный контроль над позицией элементов относительно их исходного положения.

- Смещает элемент по горизонтали (ось X) и вертикали (ось Y).
- **Первый параметр** - смещение по горизонтали (положительное значение - вправо, отрицательное - влево).
- **Второй параметр** - смещение по вертикали (положительное значение - вниз, отрицательное - вверх).
- **Можно использовать любые единицы измерения**, включая проценты.
- Проценты рассчитываются относительно размеров самого элемента.

```
Смещение на 20px вправо и 10px вниз */
.element {
   transform: translate(20px, 10px); /*}

/* Смещение на 50% ширины элемента вправо */
.element {
   transform: translateX(50%);}

/* Смещение на 2 размера шрифта вверх */
.element {
   transform: translateY(-2em);}
```

JAVARUSH

Функции масштабирования: scale()

Она позволяет увеличивать или уменьшать элемент по горизонтали (ось X) и вертикали (ось Y), создавая эффекты увеличения, уменьшения или зеркального отражения.

Особенности:

- Если передать только один параметр, он применяется к обеим осям: scale(2) эквивалентно scale(2, 2).
- Можно использовать отрицательные значения для создания зеркального отражения:
- scale(-1, 1) - отражение по горизонтали
- scale(1, -1) - отражение по вертикали

```
.element {
   transform: scale(1.5);  /* Увеличивает элемент в
1.5 раза по обеим осям */
}


.element {
   transform: scale(0.8, 1.2); /* Уменьшает по
горизонтали, увеличивает по вертикали */
}


.element {
   transform: scale(-1, 1); /* Зеркальное отражение
по горизонтали */
}
```

JAVARUSH

Функции поворота: rotate()

Вы можете вращать элементы вокруг разных осей, добавляя глубину и перспективу вашему дизайну.

rotateX(X), rotateY(Y), rotateZ(Z)

- Вращают элемент вокруг указанной оси: X (горизонтальная), Y (вертикальная) или Z (перпендикулярная экрану).
- Принимают один параметр - угол поворота в градусах (deg), радианах (rad) или оборотах (turn).
- rotateZ() эквивалентно rotate(), то есть вращает элемент вокруг оси Z (обычное вращение на плоскости экрана).

```
.element {
   transform: rotate(45deg); /* Поворот на 45
градусов по часовой стрелке */
}

.element {
   transform: rotateX(30deg); /* Поворот на 30
градусов вокруг горизонтальной оси */
}

.element {
   transform: rotate3d(1, 0, 0, 45deg); /* Поворот
на 45 градусов вокруг оси X */
}
```

JAVARUSH

Функции наклона: skewX() и skewY()

Функции skewX() и skewY() позволяют наклонять элементы по горизонтали или вертикали, создавая эффект перспективы или искажения.

- skewX(X): Наклоняет верхнюю сторону элемента относительно нижней на заданный угол X (в градусах).
- skewY(Y): Наклоняет правую сторону элемента относительно левой на заданный угол Y (в градусах).

```
.element {
   transform: skewX(30deg); /* Наклоняет верхнюю сторону вправо на 30 градусов */
}

.element {
   transform: skewY(-15deg); /* Наклоняет правую сторону вверх на 15 градусов */
}
```

JAVARUSH

Точка отсчёта для трансформаций

Свойство transform-origin позволяет вам выбрать точку, вокруг которой будут происходить трансформации элемента, такие как повороты, масштабирование и наклоны. По умолчанию эта точка находится в центре элемента, но вы можете изменить её положение для достижения различных визуальных эффектов.

```
element {
   transform-origin: x-axis y-axis z-axis;
}
```

Примеры значений:

- transform-origin: 50% 50%; - центр элемента (значение по умолчанию)
- transform-origin: 0 0; - верхний левый угол элемента
- transform-origin: 100% 100%; - нижний правый угол элемента
- transform-origin: 50px 100px; - смещение на 50 пикселей вправо и 100 пикселей вниз от верхнего левого угла

JAVARUSH

Центрирование точки происхождения

По умолчанию трансформации применяются относительно центра элемента. Например, при повороте элемента на 45 градусов (transform: rotate(45deg);), вращение будет происходить вокруг его центра.

```
.box {
    width: 100px;
    height: 100px;
    background-color: #3498db;
    transform: rotate(45deg);
}
```

JAVARUSH

Комбинированные трансформации

CSS позволяет нам комбинировать несколько трансформаций в одном свойстве transform, что открывает двери для создания удивительных визуальных эффектов. Представьте, что вы можете одновременно перемещать, поворачивать, масштабировать и наклонять элементы, создавая настоящие шедевры!

Просто перечисляйте функции трансформации через пробел, и они будут применены к элементу последовательно, в порядке их указания.

```
element {
   transform: translate(20px, 20px)
rotate(45deg) scale(1.5) skew(10deg, 5deg);
}
```

Порядок функций в transform имеет значение! Результат будет зависеть от того, в каком порядке вы применяете трансформации.

JAVARUSH

Плавные переходы в CSS

Свойство transition - это волшебная палочка, которая оживляет ваши стили, добавляя плавные переходы между состояниями элементов.

Что можно анимировать:

- Практически любые CSS-свойства, которые могут изменяться: width, height, background-color, opacity, transform и многие другие.

```
element {
   transition: property duration timing-function delay;
}
```

- property: свойство, к которому применяется переход (например, width, height, background-color)
- duration: продолжительность перехода (например, 2s для двух секунд или 500ms для 500 миллисекунд)
- timing-function: функция тайминга, определяющая скорость изменения анимации (например, ease, linear, ease-in, ease-out, ease-in-out)
- delay: задержка перед началом перехода (например, 1s для одной секунды)

JAVARUSH

Свойства переходов

Свойство transition-property определяет, к каким свойствам элемента будут применяться переходы.

```
element {
   transition-property: property1, property2 ...;
}
```

Свойство transition-timing-function определяет скорость изменения анимации в течение времени.

```
element {
   transition-timing-function: function;
}
```

Свойство transition-duration задает продолжительность перехода.

```
element {
   transition-duration: time;
}
```

Свойство transition-delay задает задержку перед началом перехода.

```
element {
   transition-delay: time;
}
```

JAVARUSH

Функции тайминга

Функции тайминга определяют, как значение свойства изменяется в течение времени. Они позволяют создавать различные эффекты анимации.

Основные функции тайминга:

- ease: анимация начинается медленно, ускоряется к середине и замедляется к концу
- linear: анимация происходит с постоянной скоростью от начала до конца
- ease-in: анимация начинается медленно и затем ускоряется
- ease-out: анимация начинается быстро и затем замедляется
- ease-in-out: анимация начинается и заканчивается медленно, ускоряется в середине

JAVARUSH

Применение переходов

Давайте рассмотрим, как анимировать изменение ширины элемента при наведении курсора мыши.

```
.box {
    width: 100px;
    height: 100px;
    background-color: #f39c12;
    transition: width 2s ease-in-out; /* Плавное изменение ширины за 2 секунды */
}

.box:hover {
    width: 200px; /* Увеличиваем ширину при наведении */
}
```

Объяснение:

- .box: Задаем начальные стили элемента: ширина 100px, высота 100px, оранжевый фон.
- transition: width 2s ease-in-out;: Указываем, что при изменении ширины будет происходить плавный переход в течение 2 секунд с эффектом ускорения в начале и замедления в конце (ease-in-out).
- .box:hover: При наведении курсора на элемент его ширина плавно увеличивается до 200px.

JAVARUSH

Good work

https://codepen.io/TheAngularGuy/pen/yLoqgXb

JAVARUSH

Домашнее задание

Уровень 12. Трансформации. Переходы и анимация

Лекции 0-4