CSS
Трансформации

CSS - 2D Transformations (Plan)

CSS - 3D Transformations (Plan)

- Координатная система в 3D пространстве
- transform: translateZ()
- transform-style
- transform: translate3d()
- transform: rotateX()
- transform: rotateY()
- transform: rotateZ()
- perspective
- perspective-origin
- backface-visibility

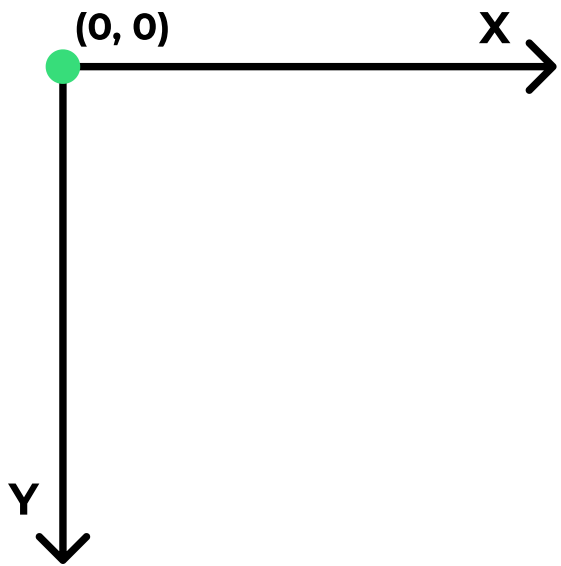
CSS-трансформации (transformations)

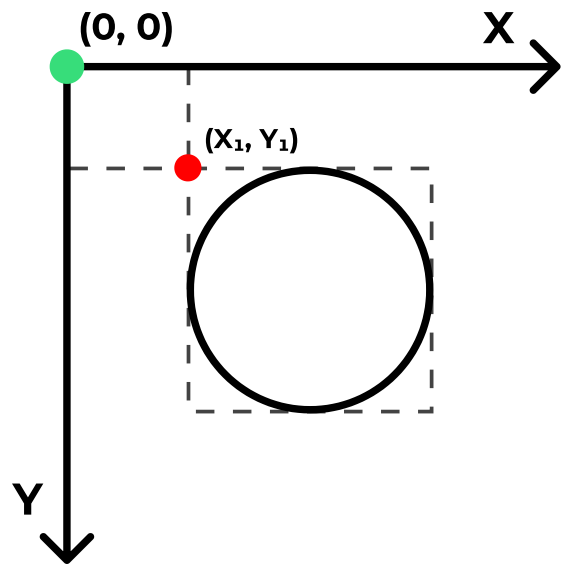
позволяют трансформировать элементы, т. е. перемещать их, вращять, наклонять и изменять масштаб, как в двумерном, так и в трехмерном пространстве

Использование CSS трансформаций

Двумерные Трансформации

Обратная система координат





Прямоугольная система координат

Свойство transform

```
transform: [функция трансформации]([значение трансформации])
```

CSS transform

transform: translateX()

задает сдвиг по горизонтали

```
selector {
  transform: translateX(10px); /* Сдвиг вправо на 10px */
}
```

TranslateX: DEMO

transform: translateX(<значения>)

- сдвиг по X – задается в "px","%","em" и др.

TranslateX

transform: translateY()

задает сдвиг по вертикали

```
selector {
  transform: translateY(15px); /* Сдвиг вниз на 15px */
}
```

TranslateY: DEMO

transform: translateY(<значения>)

– сдвиг по Y – задается в "px", "%", "em" и др.

TranslateY

transform: translate()

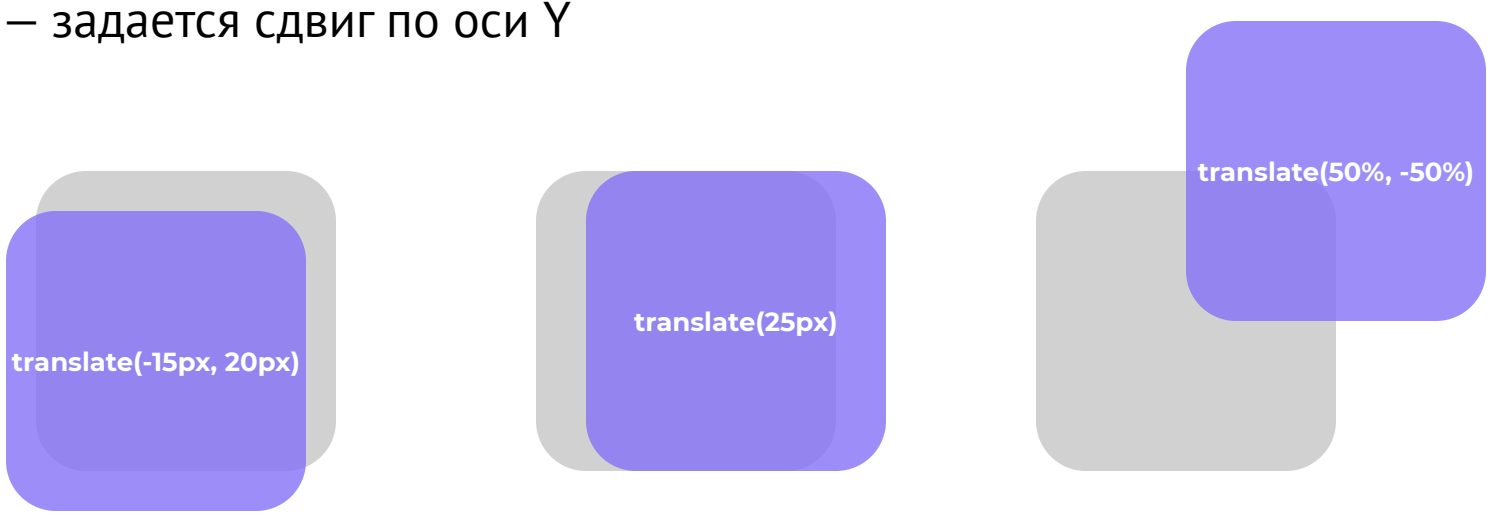
задает сдвиг по горизонтали и по вертикали

```
selector {
  transform: translate(15px, 25%);
}
```

Translate: DEMO

transform: translate(O_x, O_y)

- O_x – задается сдвиг по оси X
- O_y – задается сдвиг по оси Y



Translate

transform: scaleX()

изменяет масштаб по горизонтали

```
01. selector {
02.   transform: scaleX(1.5);
03. }
```

ScaleX: DEMO

transform: scaleX(<значения>)

- Масштаб по оси X – задается коэффициентом (k)
  - Если k > 1, то элемент растягивается
  - Если k < 1, то элемент сжимается
  - Если k < 0, то элемент зеркально отразится по оси X

ScaleX

transform: scaleY()

изменяет масштаб по вертикали

```
01. selector {
02.   transform: scaleY(2);
03. }
```

ScaleY: DEMO

transform: scaleY(<значения>)

- Масштаб по оси Y – задается коэффициентом (k)
  - Если k > 1, то элемент растягивается
  - Если k < 1, то элемент сжимается
  - Если k < 0, то элемент зеркально отразится по оси Y

transform: scale()

изменяет масштаб по горизонтали и по вертикали

```
01. selector {
02.   transform: scale(1.5, 1.7);
03. }
```

Scale: DEMO

transform: scale(O_x, O_y)

- O_x — изменяет масштаб по оси X
- O_y — изменяет масштаб по оси Y

Scale

transform: rotate()

задает поворот элемента на нужный градус

```
01. selector {
02.   transform: rotate(45deg);
03. }
```

Rotate: DEMO

transform: rotate(<значения>)

- поворот – задается в "deg", "rad", "grad" и "turn"
  - Если угол положительный, то элемент поворачивается по часовой стрелке
  - Если угол отрицательный, то элемент поворачивается против часовой стрелки

Rotate

transform: skewX()

задает наклон элемента относительно оси X

```
01. selector {
02.   transform: skewX(15deg);
03. }
```

SkewX: DEMO

transform: skewX(<значения>)

- наклон – задается в "deg", "rad", "grad" и "turn"
  - Если угол положительный, то элемент наклоняется против часовой стрелки
  - Если угол отрицательный, то элемент наклоняется по часовой стрелке

SkewX

transform: skewY()

задает наклон элемента относительно оси Y

```
selector {
  transform: skewY(20deg);
}
```

SkewY: DEMO

transform: skewY(<значения>)

- наклон – задается в "deg", "rad", "grad" и "turn"
  - Если угол положительный, то элемент наклоняется по часовой стрелке
  - Если угол отрицательный, то элемент наклоняется против часовой стрелки

SkewY

transform: skew()

задает наклон элемента относительно оси X и оси Y

```
01. selector {
02.   transform: skew(25deg, -20deg);
03. }
```

Skew: DEMO

transform: skew(O_x, O_y)

- O_x – изменяет наклон по оси X
- O_y – изменяет наклон по оси Y

Skew

Свойство transform-origin

устанавливает координаты точки (anchor point), относительно которой будет происходить трансформация элемента

Transform-Origin

transform-origin

```
selector {
  transform-origin: center; /* По умолчанию в центре */
}
```

transform-origin: DEMO

transform-origin: O_x O_y O_z

- O_x — координата по оси X

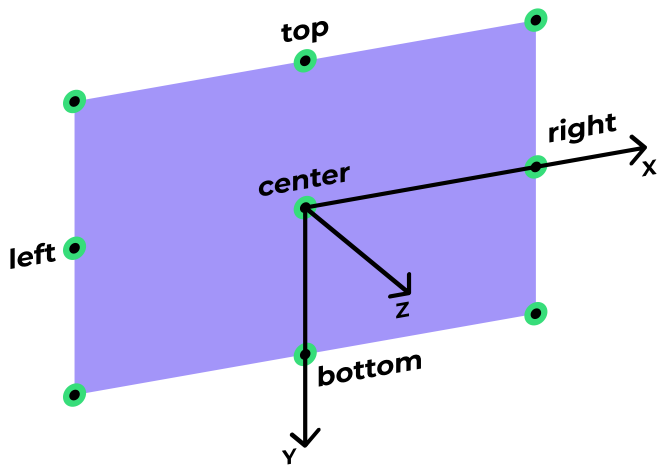
  <длина> | <проценты> | left | center | right

- O_y — координата по оси Y

  <длина> | <проценты> | top | center | bottom

- O_z — координата по оси Z

  только <длина>

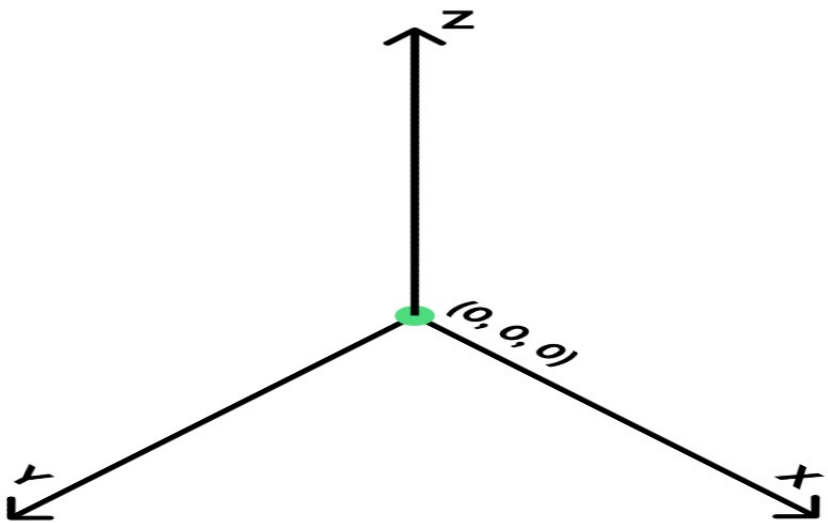


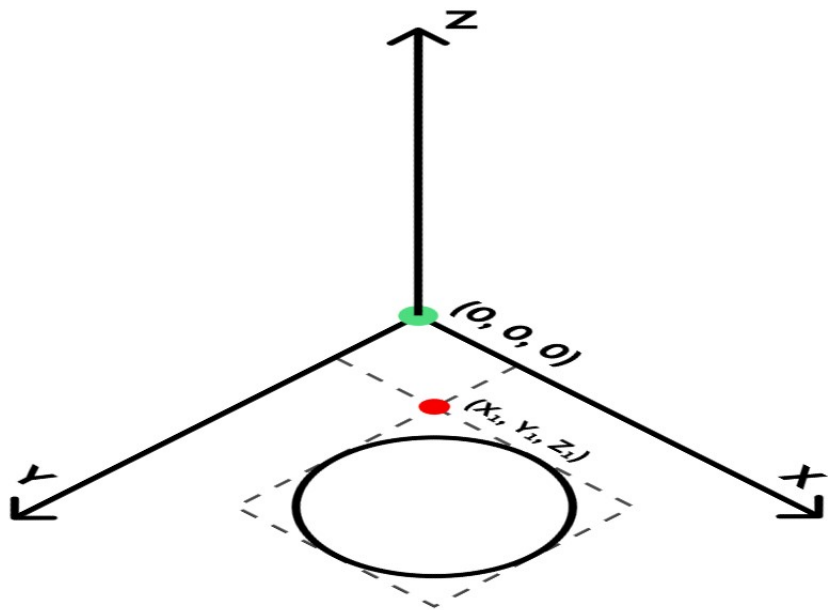
Transform-Origin

transform-origin + rotate(25deg)

Трехмерные Трансформации

Координатная система в трехмерном пространстве





transform: translateZ()

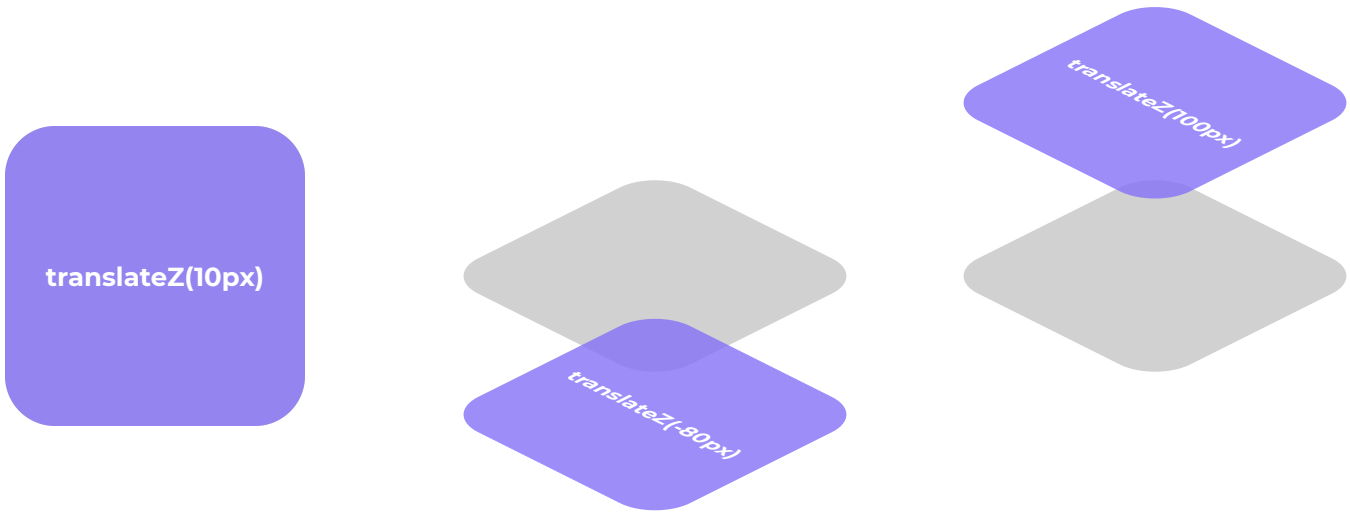
задает сдвиг по оси Z

```
01. selector {
02.   transform: translateZ(20px);
03. }
```

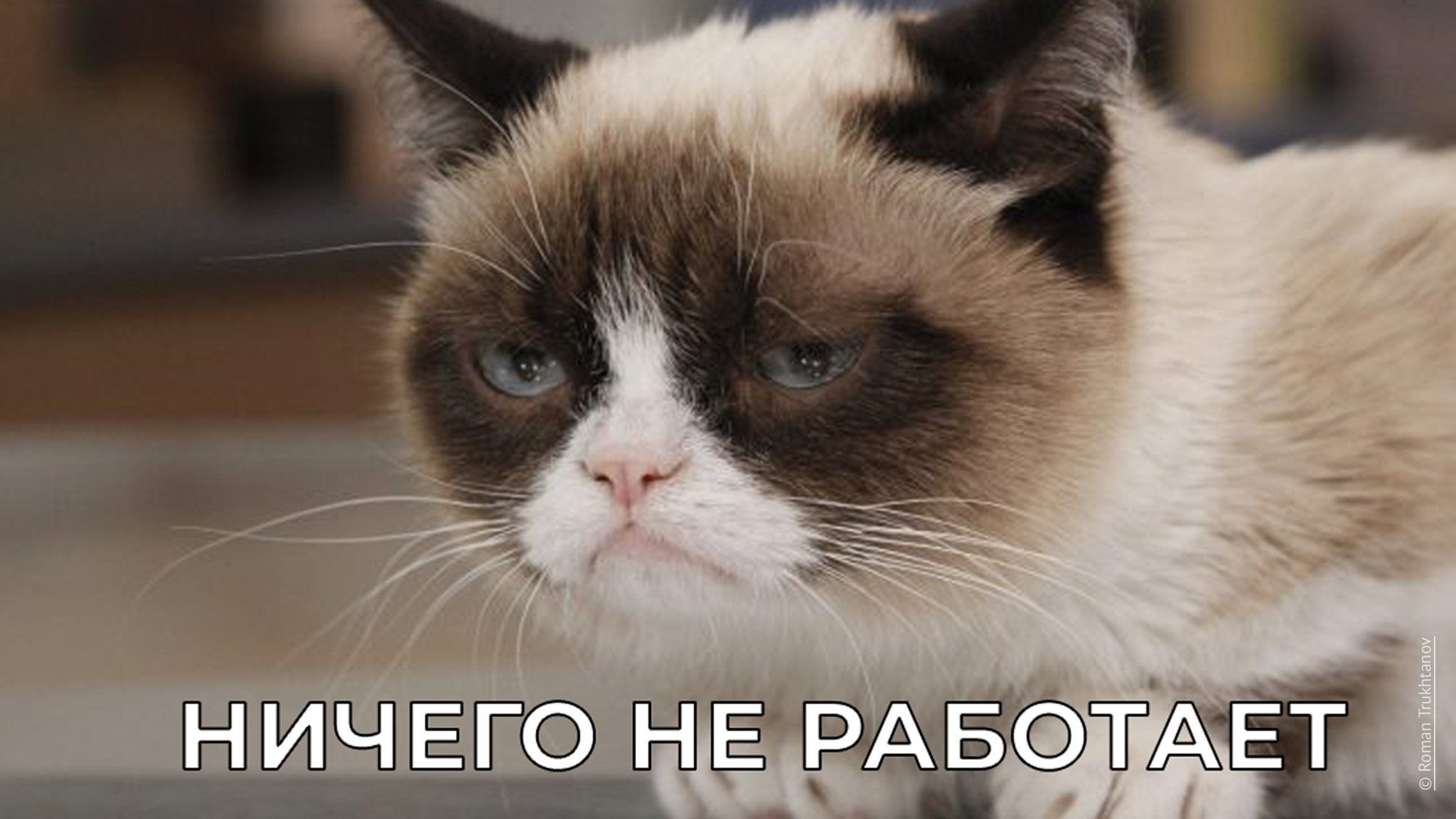
TranslateZ: DEMO

transform: translateZ(<значения>)

- сдвиг по Z – задается в "px", "em" и др. (но не в "%")



TranslateZ



transform-style

определяет в каком пространстве будут располагаться дочерние элементы

```
selector {
  transform-style: preserve-3d;
}
```

Transform-Style: DEMO

transform-style: <значения>

- `flat` — дочерние элементы будут располагаться в плоскости
- `preserve-3d` — дочерние элементы будут располагаться в 3D пространстве

Transform-Style

transform: translate3d()

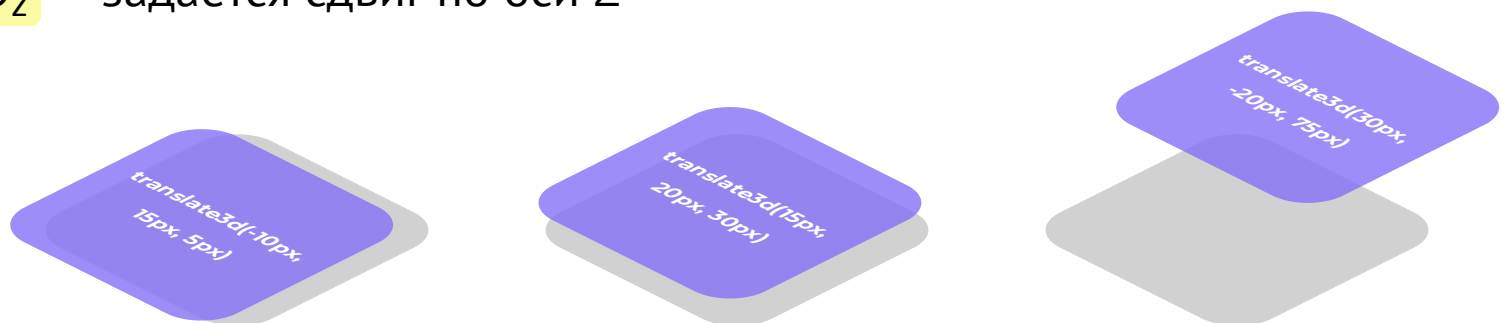
задает сдвиг по осям X, Y, Z

```
01. selector {
02.   transform: translate3d(10px, -15px, 20px);
03. }
```

Translate3D: DEMO

transform: translate3d(O_x, O_y, O_z)

- O_x – задается сдвиг по оси X
- O_y – задается сдвиг по оси Y
- O_z – задается сдвиг по оси Z



Translate3D

transform: rotateX()

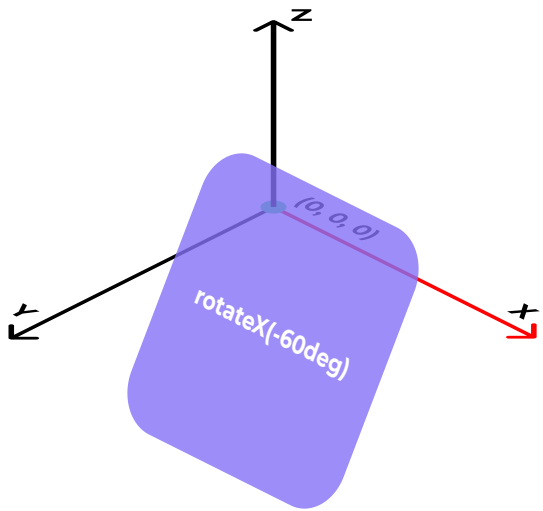
задает вращение по оси X

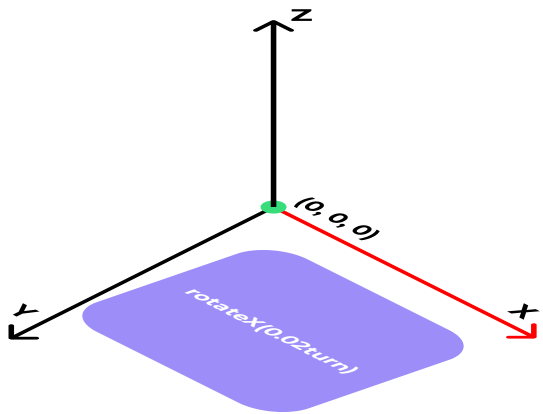
```
01. selector {
02.   transform: rotateX(30deg);
03. }
```

RotateX: DEMO

transform: rotateX(<значения>)

- поворот — задается в "deg", "rad", "grad" и "turn"





RotateX

transform: rotateY()

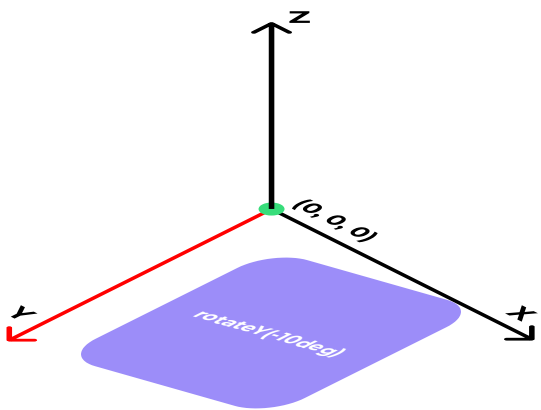
задает вращение по оси Y

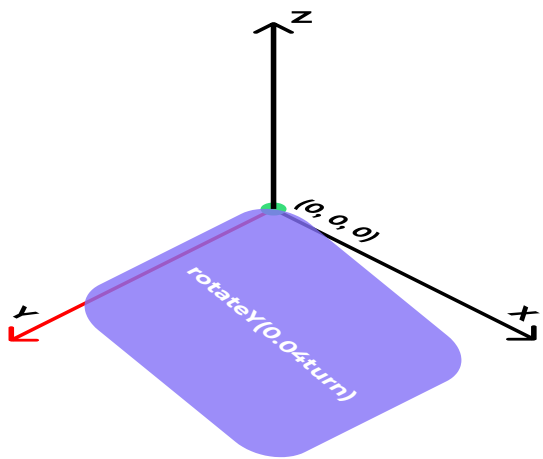
```
01. selector {
02.   transform: rotateY(-45deg);
03. }
```

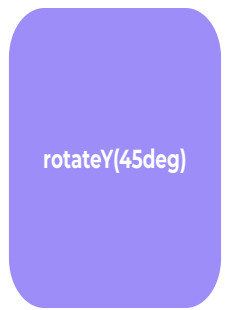
RotateY: DEMO

transform: rotateY(<значения>)

– поворот – задается в "deg", "rad", "grad" и "turn"







RotateY

transform: rotateZ()

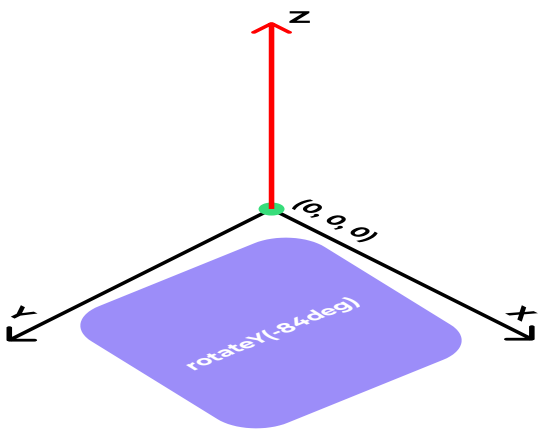
задает вращение по оси Z

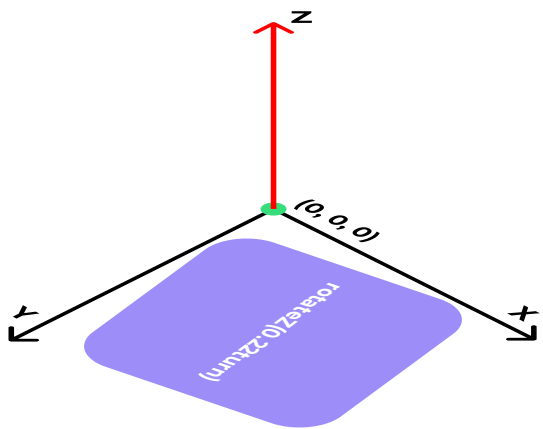
```
01. selector {
02.   transform: rotateZ(90deg);
03. }
```

RotateZ: DEMO

transform: rotateZ(<значения>)

– поворот – задается в "deg", "rad", "grad" и "turn"





rotateZ(90deg)

RotateZ

perspective

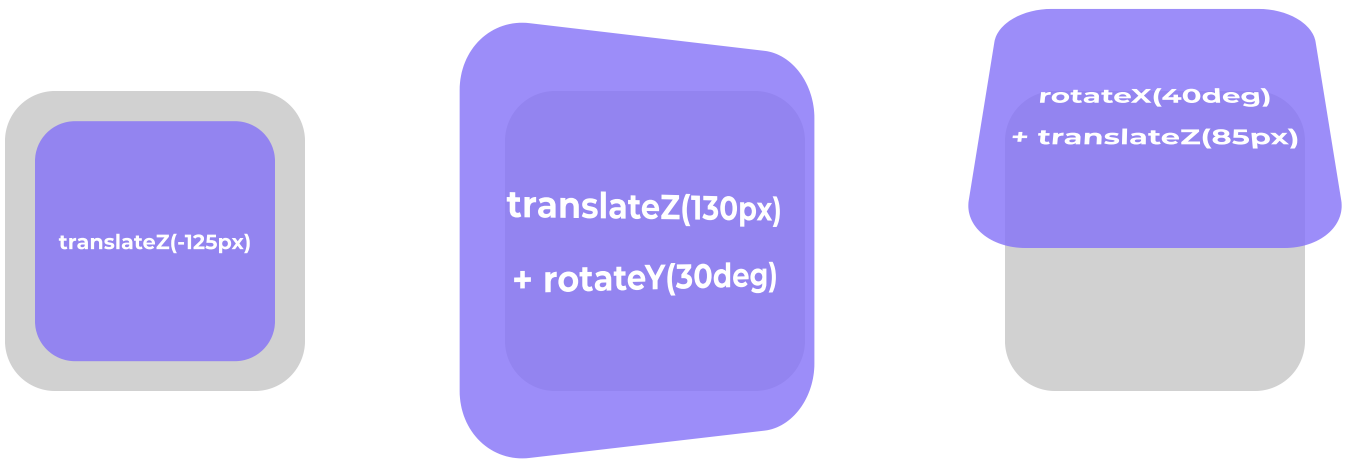
позволяет задать эффект глубины дочерним элементам

```
01. selector {
02.   perspective: 1000px;
03. }
```

Perspective: DEMO

perspective: <значения>

- none – эффект глубины не будет применен
- <длина> – задается в "px", "em" и др.



Perspective

perspective-origin

задает координаты точки схождения, т. е. условно то место, куда смотрит наблюдатель

```
selector {
  perspective-origin: left top;
}
```

Perspective-Origin: DEMO

perspective-origin: O_x O_y

- O_x — координата по оси X

  <длина> | <проценты> | left | center | right

- O_y — координата по оси Y

  <длина> | <проценты> | top | center | bottom

Perspective-Origin

perspective-origin + translateZ(-100px)

backface-visibility

определяет видимость обратной стороны элемента, когда он повернут к пользователю

```
selector {
  backface-visibility: hidden;
}
```

Backface-Visibility: DEMO

backface-visibility: <значения>

- visible — обратная сторона будет видна
- hidden — обратная сторона не будет видна

default **visible** **hidden**

Backface-Visibility