Transform:

Transform Property: A CSS command that applies 2D or 3D transformations to **elements**. Such transformations can either be rotate, scale, move, skew, etc.

1. **Syntax:**

```
.front {
  transform: translateZ(50px);
  transform: rotateY(180deg) translateZ(50px);
}
```

2. **2D Transformation Methods:**
   A. **Rotate**: Rotates the element at an angle. When the degrees are positive, the element will rotate clock-wise. When the degrees are negative, they rotate counter clock-wise. You denote the rotation with degrees, denoted by "deg."

```
transform: rotate(100deg, 180deg);
```

   The **example above** would rotate the element 100 degrees clock-wise about the x-axis and 180 degrees about the y-axis.

   Demo: https://www.w3schools.com/css/css3_2dtransforms.asp

   B. **Translate**: Moves an element from its set origin to another position along the x- and y- axes. You use pixels, or other similar standards of measure.

```
transform: translate(moves along x-axispx, moves along y-axispx);
```

   Demo: https://www.w3schools.com/css/tryit.asp?filename=trycss3_transform_translate

   C. **Scale**: Increases or decreases the size of an element according to the parameters given to the width (x-axis) and height (y-axis).

```
transform: scale(moves along x-axis, moves along y-axis);
```

   Demo: https://www.w3schools.com/css/tryit.asp?filename=trycss3_transform_scale

   D. **skew**: Skews an element along the x- and y-axes by a given angle. Note: The element MUST be skewed at an angle (deg), radian (rad), or gradian (grad) denoted by the "deg" inside of the parameter.

```
transform: skew(100deg, 180deg);
```

   You can also skew elements along just their x- or y- axes:

I. **skewX**: skews an element along the x-axis by a given angle. Note: The element MUST be skewed at an angle, denoted by the "deg" inside of the parameter.

```
transform: skewX(50deg);
```

Demo:
https://www.w3schools.com/css/tryit.asp?filename=trycss3_transform_skewx

II. **skewY**: skews an element along the y-axis by a given angle. The element is also skewed in degrees denoted by the "deg."

```
transform: skewY(100deg);
```

Demo:
https://www.w3schools.com/css/tryit.asp?filename=trycss3_transform_skewy

E. **matrix**: Combines **ALL** of the 2D methods into one. The matrix method takes six parameters that set the dimensions for rotate, scale, translate, and skew.
The parameters are as follow:

```
transform: matrix(1, 0, 0.5, 1, 150, 0);
```

Explanation: The parameters are as follow:
matrix(scaleX,skewY,skewX,scaleY,translateX,translateY)

PRO TIP: This website takes all of the transformations that you would like to implement, and spits out a matrix for you that you can add into your CSS so that you don't have to do it manually!
https://meyerweb.com/eric/tools/matrix/

Summary:

Function	**Description**
matrix(*n*,*n*,*n*,*n*,*n*,*n*)	Defines a 2D transformation, using a matrix of six values
translate(*x*,*y*)	Defines a 2D translation, moving the element along the X- and the Y-axis
translateX(*n*)	Defines a 2D translation, moving the element along the X-axis
translateY(*n*)	Defines a 2D translation, moving the element along the Y-axis
scale(*x*,*y*)	Defines a 2D scale transformation, changing the elements width and height
scaleX(*n*)	Defines a 2D scale transformation, changing the element's width
scaleY(*n*)	Defines a 2D scale transformation, changing the element's height
rotate(*angle*)	Defines a 2D rotation, the angle is specified in the parameter
skew(*x-angle*,*y-angle*)	Defines a 2D skew transformation along the X- and the Y-axis
skewX(*angle*)	Defines a 2D skew transformation along the X-axis
skewY(*angle*)	Defines a 2D skew transformation along the Y-axis

Transform-Origin Property: A CSS property that allows you to change the position of already transformed elements. So you use the “transform-origin” property as a second layer of transformation to elements. Think of it as transformation squared! NOTE: This property **MUST** be used **in conjunction** with the “transform” property.

1. **Syntax**:

```
transform-origin: x-axis y-axis z-axis|initial|inherit;
```

2. **Transform-Origin Property Values**

   A. Moves along the **x-axis**:
   These are the possible property values for an origin-property transformation along the x-axis.
      I. Left
      II. Center
      III. Right
      IV. Length
      V. %

   B. Moves along the **y-axis**:
   These are possible property values for an origin-property transformation along the y-axis.
      I. Top
      II. Center
      III. Bottom
      IV. Length
      V. %

   C. Moves along the **z-axis**:
   Defines where the view is placed at the z-axis (for 3D transformations). Possible values:
      I. length

   D. **Initial**:
      I. The initial property sets the elements CSS property to its default value.

   E. **Inherit**:
   The inherit property stipulates that the CSS element inherit its parent property.

   F. **Examples**:

```
transform-origin: 50% 50% 0;
```

   Transforms the element to the center of the container: 50% to the right, 50% to the left.

```
transform-origin: 20px 70%;
```

   Transforms the element 20px to the right and 70% south of its original position.

```
transform-origin: top right;
```
Transforms the element about the top and right point of the element.
```
transform-origin: center bottom;
```
Transforms the element about the center-bottom point of the element.

G. **Demo**:
https://cssreference.io/property/transform-origin/
Master Demo:
https://css-tricks.com/almanac/properties/t/transform-origin/

3D Transformation Methods:

1. **Rotate**: Rotates the element about the x- y- or z- axes at an angle, radian, or gradian. When the degrees are positive, the element will rotate clock-wise. When the degrees are negative, they rotate counter clock-wise. For the z-axis, when the units of measure are **positive**, the element will point **towards you**, when the units are **negative**, the element will point **away from you**.
   A. **rotateX(angle):** Specifies a 3D rotation about the x-axis. Think about those old-school doors that used to have the mail slots or window blinds when you open or close them. They "transform" about the x-axis when you open the blinds, or open the mail slot in the door to drop the mail in.
   B. **rotateY(angle):** specifies a 3D rotation about the y-axis. Think about a door swinging open, it rotates about its y-axis in relation to its origin(the door hinges).
   C. **rotateZ(angle):** specifies a 3D rotation about the z-axis.
      I. Let's talk about the **z-axis**. 3D transformations also have a z-axis, **unlike** 2D transformations. The z-axis is a lot like Samara in "The Ring" when she comes out of the television screen. The z-axis "goes through" the screen and re-defines our perspective of the element. The z-axis points towards or away from you. AKA the depth.
   D. **rotate3d(x,y,z,angle):** rotate3D takes **4 properties.** The first 3 are the x- y- and z- properties, and the last is the degree of rotation for said properties.
   E. **Syntax:**

```
transform: rotateX(210deg);

transform: rotateY(67deg);

transform: rotateZ(150deg);

transform: rotate3d(.1,.2,.5, 250deg)
```

Note: The .1, .2, and .5 denote an axis of rotation about an element's vector. You learn more about the matrices vectors in computer science but in layman's terms, these numbers represent the angles of rotation about the "lines" of an element. These measurements are expressed in radians, but do not require the "rad" designation. **They can only be positive and range from 0 – 1**.

H. **Live Demo**:
https://developer.mozilla.org/en-US/docs/Web/CSS/transform-function/rotate3d

Try It Yourself!
https://dev.opera.com/articles/understanding-3d-transforms/rotate3d.html

2. **Translate**: Moves an element from its set origin to another position along the x-, y- and z- axes. The unites of measurement for "translate" are px, rem, or em, etc.
   A. translateX(measurement): Denotes a 3D translation about the x-axis.
   B. translateY(measurement): Denotes a 3D translation about the y-axis.
   C. translateZ(measurement): Denotes the 3D translation about the z-axis.
   D. translate3d(x,y,z): Allows the elements to translate about the x- y- and z- axes simultaneously.

   E. **Syntax**:

```
transform: translateX(300px);

transform: translateY(200px);

transform: translateZ(230px);

transform: translate3d(200px, 70%, 5em);
```

   F. **Demo**: https://developer.mozilla.org/en-US/docs/Web/CSS/transform-function/translate3d

3. **Scale**: Increases or decreases the size of an element according to the parameters given to the width (x-axis) and height (y-axis). The units of measurement for scale are dependent upon the units of measurement of the original specified conditions. The units of measurement are denoted by non-integer and integer values inside of the parameters.
   A. **scaleX()**: Resizes an element about the x-axis. AKA Makes the element wider.
   B. **scaleY()**: Resizes an element about the y-axis. AKA elongates the element about the y-axis.
   C. **scaleZ()**: Resizes an element about the z-axis. AKA changes the dimension of the element from the perspective of the screen.
      1. **Demo**: https://developer.mozilla.org/en-US/docs/Web/CSS/transform-function/scaleZ

   D. **scale3d(x,y,z)**: Resizes the element simultaneously about the x-, y-, and z- axes.
      I. **Demo**: https://developer.mozilla.org/en-US/docs/Web/CSS/transform-function/scale3d

   E. **Syntax**:

```
transform: scaleX(.25);

transform: scaleY(2);

transform: scaleZ(1);

transform: scale3d(2,1,3);
```

Note: The units of measurement for scale are dependent upon the units of measurement of the original specified dimensions. In the above examples, the element that takes the property of scaleX is scaled down to ¼ **of its original size** along the x-axis. In the second example, the element’s scale is increased to **twice its original size** along the y-axis.

   F. **Demo**: https://developer.mozilla.org/en-US/docs/Web/CSS/transform-function/scale3d

4. **Perspective**: Defines a perspective view for a 3D transformed element. Essentially, “transform:perspective()” represents the distance between the user and the z-axis at its default origin.
   A. **Measurement**: A **positive value** in the parameter makes the element **appear closer** to the **user**. The smaller the unit of measurement, the **closer the perspective**. The measurement takes px, rem, em, cm, in, etc. units of measurement.
   Note: The perspective parameter cannot take negative values or a value of “0.”

   B. **Syntax**:

```
transform: perspective(0);
transform: perspective(800px);
transform: perspective(23rem);
transform: perspective(6.5cm);
```

   C. **Live Demo**:
   https://developer.mozilla.org/en-US/docs/Web/CSS/transform-function/perspective
   Try it yourself!
   https://codepen.io/pen/?&editable=true

5. **matrix3d**: Combines **ALL** of the 3D methods into one. The matrix method takes sixteen parameters that set the dimensions and transformations for the element.
   The parameters are as follow:

```
transform: matrix3d(a1, b1, c1, d1, a2, b2, c2, d2, a3, b3, c3, d3, a4, b4, c4, d4)
```

   Explanation: The matrix3d parameters are too cumbersome to apply manually. See tip below.

   PRO TIP: This website takes all of the transformations that you would like to implement, and spits out a matrix for you that you can add into your CSS so that you don’t have to do it manually!
   https://meyerweb.com/eric/tools/matrix/

3D Transformation Properties:
1. **Transform**:
2. **Transform-Origin Property Values**: See 2D explanation and below. **MUST** be used **in conjunction** with the “Transform property.”

A. Moves along the **z-axis**:
Defines where the view is placed at the z-axis (for 3D transformations). Possible values:
   I. length

3. **Transform-Style**: Specifies how **nestled elements** (aka children) are rendered in the 3D space, or whether they are "flattened". This property **MUST** be used **in conjunction** with the "Transform" property.

   A. **flat:** Specifies that children elements **will NOT preserve** the 3D position (default). AKA the children elements are no longer "stacked" according to the translate values and instead appear the way that elements appear by default on HTML page.
      1. **Syntax**:

```
transform-style: flat;
transform: rotateY(245deg);
```

   B. **preserve-3d:** Specifies that the children elements **WILL preserve** the 3D position.
      I. **Syntax**:

```
transform-style: preserve-3d;
transform: rotateY(245deg);
```

   C. **Live Demo**:
   https://tympanus.net/codrops-playground/SaraSoueidan/abqrPXfg/editor
   https://developer.mozilla.org/en-US/docs/Web/CSS/transform-style
   *The last demo is a 2D simulation of a 3D rendering.

4. **Perspective**: The perspective property is used to give a 3D-positioned element some perspective. It defines **how far the object is away from the user**. A **lower value** will result in a **more intensive 3D effect** than a higher value. When defining the perspective property for an element, it is the **CHILD elements** that **get the perspective view**, NOT the element itself.
   A. **Difference between "transform:perspective()" VS the perspective property**: The main difference between the "transform:perspective()" and the perspective property is that the **former** gives **an element** depth, while the **latter** creates a **3D space (based on the z-axis)** of all of the nestled children.

   B. **Methods**:
      a. **length**: How far the element is placed from view.
      b. **none**: Default value. Same as "0." No set perspective.
      c. **inherit**: Inherits property from the parent element.

   C. **Syntax**:

```
/* Keyword value */
```

```
perspective: none;
/* <length> values */
perspective: 20px;
perspective: 3.5em;
/* Global values */
perspective: inherit;
perspective: initial;
perspective: unset;
```

D. **Live Demo**:
https://developer.mozilla.org/en-US/docs/Web/CSS/perspective
Try it yourself!
https://www.w3schools.com/cssref/trycss3_perspective_inuse.htm

5. **Perspective-Origin**: This property **MUST** be used in conjunction with the “Perspective” property.

A. **Methods**:
   a. **x-axis**: Defines where the view is place on the x-axis.
      Possible values:
      i. left:
      ii. center:
      iii. right:
      iv. length:
      v. %:

   b. **y-axis**: Defines where the view is placed on the y-axis.
      i. Top:
      ii. Center:
      iii. Bottom:
      iv. Length:
      v. %:

   c. **initial**: Sets the perspective-origin property to its initial value.
   d. **inherit**: Inherits the property from the parent value.

B. **Syntax**:

```
/* One-value syntax for the x-axis */
perspective-origin: left;
perspective-origin: center;
perspective-origin: right;
perspective-origin: 10px;
perspective-origin: 70%;
```

```
/* One-value syntax for the y-axis */
perspective-origin: top;
perspective-origin: center;
perspective-origin: bottom;
perspective-origin: 10px;
perspective-origin: 70%;
/* Two-value syntax */
perspective-origin: x-position y-position;
perspective-origin: left top;
perspective-origin: 0% 0%;
perspective-origin: 60% 70%;
etc.
/* When both x-position and y-position are keywords,
(aka top left) the following is also valid */
perspective-origin: y-position x-position;
/* Global values */
perspective-origin: inherit;
perspective-origin: initial;
perspective-origin: unset;
```

C. **Live Demo**:
https://css-tricks.com/almanac/properties/p/perspective-origin/
*Jess: Scroll to the bottom – look for animation.
Try it yourself!
https://www.w3schools.com/cssref/trycss3_perspective-origin_inuse.htm

6. **Backface-Visibility**: Defines whether or not an element(s) will be visible when facing the user.
    A. **Methods**:
        a. **visible**: the element will **always** be visible when facing the screen (default).
        b. **hidden**: the element is **not** visible when facing the screen.
        c. **inherit**: the element will get its property from the parent element.
        d. **initial**: sets the property to its **default**, which is **visible**.

    B. **Syntax**:

```
backface-visibility: hidden;
backface-visibility: visible;
backface-visibility: inherit;
backface-visibility: initial;
```

C. **Live Demo**:
https://developer.mozilla.org/en-US/docs/Web/CSS/backface-visibility