**Animation and effects cheat sheet**

**Transform property**

**Syntax**

transform: transform function-values

**Example**

.sample-class {

    transform: rotate(60deg);

}

**Keyword-value type: none**

.sample-class {

    transform: none;

}

**Function-value type: matrix()**

Variations: matrix(), matrix3d()

.sample-class {

     transform: matrix(1.0, 2.0, 3.0, 4.0, 5.0, 6.0);

}

**Function-value type: rotate(deg)**

Variations: rotate(), rotate3d(), rotateX(), rotate(), rotateZ()

.sample-class {

    transform: rotate3d(3,2,1, 100deg);

}

Note: In rotate3d(), the respective values represent x, y, z co-ordinate and degree of rotations

**Function-value type: translate(x,y)**

Variations: translate(), translate3d(), translateX(), translateY(), translateZ()

.sample-class {

    transform: translate3d(10px, 20px, 30px);

}

Note: In translate3d(), the respective values represent translation along the x, y, z co-ordinates

**Function-value type: scale(factor)**

Variations: scale(), scale3d(), scaleX(), scaleY(), scaleZ()

.sample-class {

    transform: scale3d(2, 1, 0.3);

}

Note: In scale3d(), the respective values represent scaling times along the x, y, z co-ordinates

**Function-value type: skew(deg, deg)**

Variations: skew(), skewX(), skewY()

.sample-class {

    transform: skew(100deg);

}

**Global value types:**

}

.sample-class {

    transform: initial;

}

.sample-class {

    transform: revert;

}

.sample-class {

    transform: revert-layer;

}

.sample-class {

    transform: unset;

}

**Multiple transform over the same element**

**Syntax**

Transform can be applied for rotate(), scale() and translate() that can be listed together. Each of these properties can have their own values and the actions will give a combined effect.

**Example**

.sample-class {

    transform: rotate(45deg) scale(1.5) translate(45px);

}

Additional property under transform:transform-origin

Determines the anchor point for the centering of transform.

**Example**

.sample-class {

    transform-origin: 10px 10px;

}

.sample-class {

    transform-origin: right bottom;

}

**Transition property**

**Transition shorthand**

Transition shorthand has four following sub-properties, each of which can also be individually defined.

* transition-property
* transition-duration
* transition-timing-function
* transition-delay

You have to list the values without naming them individually. Values skipped will be assigned their default values.

**Syntax**

transition: property duration timing-function delay;

**Example**

transition: margin-left2s ease-in-out 0.5s;

**Animations and @keyframes**

**animation property:**

**Syntax**

animation: *name duration timing-function delay iteration-count direction fill-mode play-state*;

**Example**

.sample-class {

    animation: none 2 ease 0.5 4 normal none running;

}

The animation property is a shorthand for the sub-properties below:

animation-timing-function

animation-delay

animation-iteration-count

animation-direction

animation-fill-mode

animation-play-state

The values not mentioned are given default values.

Animation-name property is used to tie-in the @keyframes rule.

**@keyframes**

**Syntax**

@keyframes mymove {

  from {property: value}

  to { property: value }

}

**Example**

@keyframes animation-name {

    from {bottom: 0px;}

    to {bottom: 100px;}

}

Percentage denotes the timing of the animation.

**Alternative syntax**

@keyframes animation-name {

/\* declare actions here \*/

}

**Example**

@keyframes animation-name {

    0%,100%{

        background-color: blue;

    }

    50% {

        background-color: green;

    }

}

**Multiple animations**

Works the same as regular animation, multiple rules can be set.

#some-class{

    animation: animation-a 2s linear infinite alternate,

        animation-b 3s ease infinite alternate;

}