CSS Units Ultimate Cheatsheet

- `px` : Absolute Length
- `rem` : Relative to the `font-size` of the root Element
- `em` : Relative to the `font-size` of the Element
- `%` : Relative to the parent Element
- `vw` : Relative to the viewport's width, `1vw` = `1%` * viewport's width
- `vh` : Relative to the viewport's height, `1vh` = `1%` * viewport's height
- `vmin` : Relative to the viewport's smaller dimension, `1vmin` = min(`1vh` , `1vw`)
- `vmax` : Relative to the viewport's larger dimension, `1vmax` = max(`1vh` , `1vw`)
- `ch` : Relative to the width of the glyph "0" of the element's font
- `in` : Inches `1in` = `2.54cm` = `96px`
- `pc` : Picas `1pc` = `1in` / `6` = `16px`
- `pt` : Points `1pt` = `1in` / `72` = `1.333px` (approximately)
- `cm` : Centimeters `1cm` = `1in` / `2.54` = `37.8px` (approximately)
- `mm` : Millimeters `1mm` = `1cm` / `10` = `3.78px` (approximately)

CODING.STELLA

CSS Flex Ultimate Cheatsheet

CONTAINER {PARENT} PROPERTIES

DISPLAY

Enables Flex For All Children

display: flex

display: inline-flex

JUSTIFY-CONTENT

Attempts to distribute extra space on main axis

flex-start

flex-end

center

space-between

space-evenly

space-around

ALIGN-ITEMS

Determine how items are laid out on the cross-axis.

flex-start

flex-end

center

baseline

stretch

ALIGN-CONTENT

Only has an effect with more than one line of content.

flex-start

flex-end

center

space-between

space-evenly

space-around

stretch

FLEX-DIRECTION

Establishes the main axis.

flex-direction: row

flex-direction: row-reverse

flex-direction: column

flex-direction: column-reverse

FLEX-WRAP

Wraps items if they can't be made to fit on one line.

flex-wrap: no-wrap

flex-wrap: wrap

flex-wrap: wrap-reverse

CSS Flex Ultimate Cheatsheet

ITEM {CHILDREN} PROPERTIES

FLEX-GROW

Allows you to determine how each child is allowed to grow as a part of a whole.

flex-grow: 1;
(Applied to all items)

flex-grow: (1, 2 & 3)

FLEX-BASIS

Define the size of an element before remaining space is distributed.

first item 20%;
second item 40%

FLEX-SHRINK

Allows an item to shrink if necessary. Only really useful with a set size or flex-basis.

both want to be 100% wide, 2nd item has flex-shrink: 2

ALIGN-SELF

Sets alignment for individual item.

3rd item has
align-self: flex-end

ORDER

The order property specifies the order of the flex items.

order: -1; on 3rd item

CODING.STELLA

CSS Grid Ultimate Cheatsheet

CONTAINER {PARENT} PROPERTIES

DISPLAY

Establishes a new grid formatting context for children.

display: grid

display: inline-grid

GRID-TEMPLATE

Defines the rows & columns of the grid.

grid-template-columns: 14px 14px 14px;
grid-template-rows: 14px 14px 14px;

grid-template-columns: repeat(3, 14px);
grid-template-rows: repeat(3, 14px)

grid-template-columns: 5px auto 5px;
grid-template-rows: 5px auto 5px;

grid-template-columns: 10% 10% auto;
grid-template-rows: 10% 10% auto;

GRID-GAP

Defines the size of column & row gutters

grid-gap: 14px;

grid-gap: 1px 14px;

grid-row-gap: 1px;
grid-column-gap: 14px ;

Note: You can also use gap which is very similar go grid for both flexbox & grid.

JUSTIFY-CONTENT

Justifies all grid content on row axis when total grid size is smaller than container.

justify-content: start

justify-content: end

justify-content: center

justify-content: stretch

justify-content: space-around

justify-content: space-evenly

justify-content: space-between

ALIGN-CONTENT

Justifies all grid content on column axis when total grid size is smaller than container.

align-content: start

align-content: end

align-content: center

align-content: stretch

align-content: space-around

align-content: space-evenly

align-content: space-between

GRID-AUTO-FLOW

Algorithm for automatically placing grid items that aren't explictly placed.

tells the auto-placement algorithm to fill in each row in turn, adding new rows as necessary

grid-auto-flow: row

tells the auto-placement algorithm to fill in each column in turn, adding new columns as necessary

grid-auto-flow: column

tells the auto-placement algorithm to attempt to fill in holes earlier in the grid if smaller items come up later

grid-auto-flow: column

JUSTIFY-ITEMS

Aligns content in a grid item along the row axis.

justify-items: start

justify-items: end

justify-items: center

justify-items: stretch (default)

ALIGN-ITEMS

Aligns content in a grid item along the column axis.

align-items: start

align-items: end

align-items: center

align-items: stretch (default)

CSS Grid Ultimate Cheatsheet

ITEM {CHILDREN} PROPERTIES

GRID-ROW

Determines an items row-based location within the grid.

```
grid-row-start: 1;
grid-row-end: 3;
```

```
grid-row-start: span 3
```

```
grid-row-start: 2;
grid-row-end: 4;
```

```
grid-row: 1 / 3;
```

```
grid-row: 1 / span 3;
```

GRID-COLUMN

Determines an items column-based location within the grid.

```
grid-column-start: 1;
grid-column-end: 3;
```

```
grid-column-start: span 3;
```

```
grid-column-start: 2;
grid-column-end: 4;
```

```
grid-column: 2 / 3;
```

```
grid-row: 2 / span 2;
```

GRID-ROW + GRID-COLUMN

Combining grid rows with grid columns.

```
grid-row: 1 / spa
grid-column: 1 / span 2;
```

```
grid-row: 2 / span 2;
grid-column: 2 / span 2;
```

JUSTIFY-SELF

Aligns content for a specific grid item along the row axis.

```
justify-self: start;
```

```
justify-self: end;
```

```
justify-self: center;
```

```
justify-self: stretch;
```

ALIGN-SELF

Aligns content for a specific grid item along the column axis.

```
align-self: start;
```

```
align-self: end;
```

```
align-self: center;
```

```
align-self: stretch;
```

CODING.STELLA

CSS Fliter Ultimate Cheatsheet

```
.selector {
    filter: blur("2px");
}
```

VALUE
PROPERTY
FUNCTION

Note: You can apply multiple functions but it has to be space seperated without comma.

No Filter Applied

filter: blur(2px);

filter: brightness(0.4);

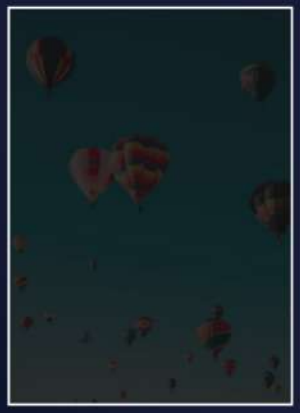
filter: contrast(200%);

filter: drop-shadow(16px red);

filter: grayscale(80%);

filter: hue-rotate(90deg);

filter: invert(85%);



filter: opacity(15%);

filter: saturate(400%);

filter: sepia(560%);

CSS Animation - Cheatsheet

ANIMATION SHORTHAND PROPERTY

```
animation: dance 300ms linear 100ms infinite alternate-reverse both reverse;
           name duration timing-function delay count direction fill-mode play-state
```

ANIMATION NAME

- Defines which animation keyframe to use.

```css
.box {
  background-color: purple;
  border-radius: 10px;
  width: 100px;
  height: 100px;

  /* Animation Name */
  animation-name: bounce;
}
```

- If no animation name is specified, no animation is played.
- If the name is specified, the keyframes matching the name will be used.

ANIMATION DURATION

- Defines how long the animation lasts.

```css
.box {
  background-color: purple;
  border-radius: 10px;
  width: 100px;
  height: 100px;

  animation-name: bounce;
  /* Animation Duration */
  animation-duration: 2s;
}
```

- The default value is **zero seconds:** the animation will simply not play.
- You can use **decimal** values in **seconds** with keyword {s}
- You can use **milliseconds** instead of seconds, with the keyword {ms}

ANIMATION TIMING FUNCTION

- Defines how the values between start & the end of the animation are calculated.

```css
.box {
  background-color: purple;
  border-radius: 10px;
  width: 100px;
  height: 100px;

  animation-name: bounce;
  animation-duration: 2s;
  /* Animation Timing Function */
  animation-timing-function: ease;
}
```

	Starts	Middle	Ends
ease:	Slow	acclearate	Slow
ease-in:	Slow	→	acclearate gradually
ease-out:	quickly	→	decelerates gradually
ease-in-out:	quickly	→	decelerates gradually
linear:	constant	constant	constant

ANIMATION DELAY

- Defines how long the animation has to wait before starting.

```css
.box {
  background-color: purple;
  border-radius: 10px;
  width: 100px;
  height: 100px;

  animation-name: bounce;
  animation-duration: 2s;
  animation-timing-function: ease;
  /* Animation Delay */
  animation-delay: 2s;
}
```

`animation-delay: 0s;`

The animation will wait **zero seconds**, and thus start right away.

`animation-delay: 1.2s;`

You can use **decimal** values in **seconds** with the keyword {s}.

`animation-delay: 2400ms;`

You can use **milliseconds** instead of seconds, with the keyword {ms}.

CODING.STELLA

CSS Animation - Cheatsheet

ANIMATION ITERATION COUNT

- Defines how many times the animation is played.

```css
.box {
  background-color: purple;
  border-radius: 10px;
  width: 100px;
  height: 100px;

  animation-name: bounce;
  animation-duration: 2s;
  animation-timing-function: ease;
  animation-delay: 2s;
  /* Animation Iteration Count */
  animation-iteration-count: 8;
}
```

`animation-iteration-count: 2s;`

You can use **integer values** to define a specific amount of times the animation will play.

`animation-iteration-count: infinite;`

By using the keyword **infinite**, the animation will play indefinitely.

ANIMATION DIRECTION

- Defines in which direction the animation is played.

```css
.box {
  background-color: purple;
  border-radius: 10px;
  width: 100px;
  height: 100px;

  animation-name: bounce;
  animation-duration: 2s;
  animation-timing-function: ease;
  animation-delay: 2s;
  animation-iteration-count: 8;
  /* Animation Direction */
  animation-direction: reverse;
}
```

`animation-direction: normal;`

Played **forwards**. When it reaches the end, it starts over at the first keyframe.

`animation-direction: reverse;`

Played **backwards**: begins at the last keyframe, finishes at the first keyframe.

`animation-direction: alternate;`

Played **forwards** first, then **backwards**.

`animation-direction: alternate-reverse;`

Played **backwards** first, then **forwards**.

ANIMATION FILL MODE

- Defines what happens before an animation starts and after it ends.

```css
.box {
  background-color: purple;
  border-radius: 10px;
  width: 100px;
  height: 100px;

  animation-name: bounce;
  animation-duration: 2s;
  animation-timing-function: ease;
  animation-delay: 2s;
  animation-iteration-count: 8;
  animation-direction: reverse;
  /* Animation Fill mode */
  animation-fill-mode: backwards;
}
```

`animation-fill-mode: none;`

The element is set to its default state before the animation **starts**, and returns to that default state after the animation **ends**.

`animation-fill-mode: forwards;`

The last styles applied at the end of the animation are retained **afterwards**.

`animation-fill-mode: backwards;`

The animation's styles will already be applied before the animation actually **starts**.

`animation-fill-mode: both;`

The styles are applied before and after the **animation plays**.

ANIMATION PLAY STATE

- Defines if an animation is playing or not.

```css
.box {
  background-color: purple;
  border-radius: 10px;
  width: 100px;
  height: 100px;

  animation-name: bounce;
  animation-duration: 2s;
  animation-timing-function: ease;
  animation-delay: 2s;
  animation-iteration-count: 8;
  animation-direction: reverse;
  animation-fill-mode: backwards;
  /* Animation Play State */
  animation-play-state: paused;
}
```

`animation-play-state: running;`

If the **animation-duration** and **animation-name** are defined, the animation will start playing automatically.

`animation-play-state: paused;`

The animation is set **paused** at the first keyframe.

CODING.STELLA

CSS Functions - Cheatsheet

CSS functional notation is the type of CSS value that can represent more complex data types or invoke special data processing or calculations.

TRANSFORM FUNCTIONS

- matrix()
- matrix3d()
- perspective()
- rotate()
- rotate3d()
- rotatex()
- rotatey()
- rotatez()
- scale()
- scalex()
- scaley()
- scalez()
- skew()
- skewx()
- skewy()
- translate()
- translate3d()
- translatex()
- translatey()
- translatez()

MATH FUNCTIONS

- calc()
- clamp()
- max()
- min()
- abs()
- acos()
- asin()
- atan()
- atan2()
- cos()
- exp()
- hypot()
- log()
- mod()
- pow()
- rem()
- round()
- sign()
- sin()
- sqrt()
- tan()

FILTER FUNCTIONS

- blur()
- brightness()
- contrast()
- drop-shadow()
- grayscale()
- hue-rotate()
- invert()
- opacity()
- saturate()
- sepia()

COLOR FUNCTIONS

- hsl()
- hsla()
- hwb()
- lab()
- lch()
- rgb()
- rgba()
- color()
- color-mix()
- color-contrast()
- device-cmyk()

FONT FUNCTIONS

- stylistic()
- styleset()
- character-varient()
- swash()
- ornaments()
- annotation()

COUNTER FUNCTIONS

- counter()
- counters()
- symbols()

GRID FUNCTIONS

- fit-content()
- minmax()
- repeat()

REFERENCE FUNCTIONS

- attr()
- env()
- url()
- var()

IMAGE FUNCTIONS

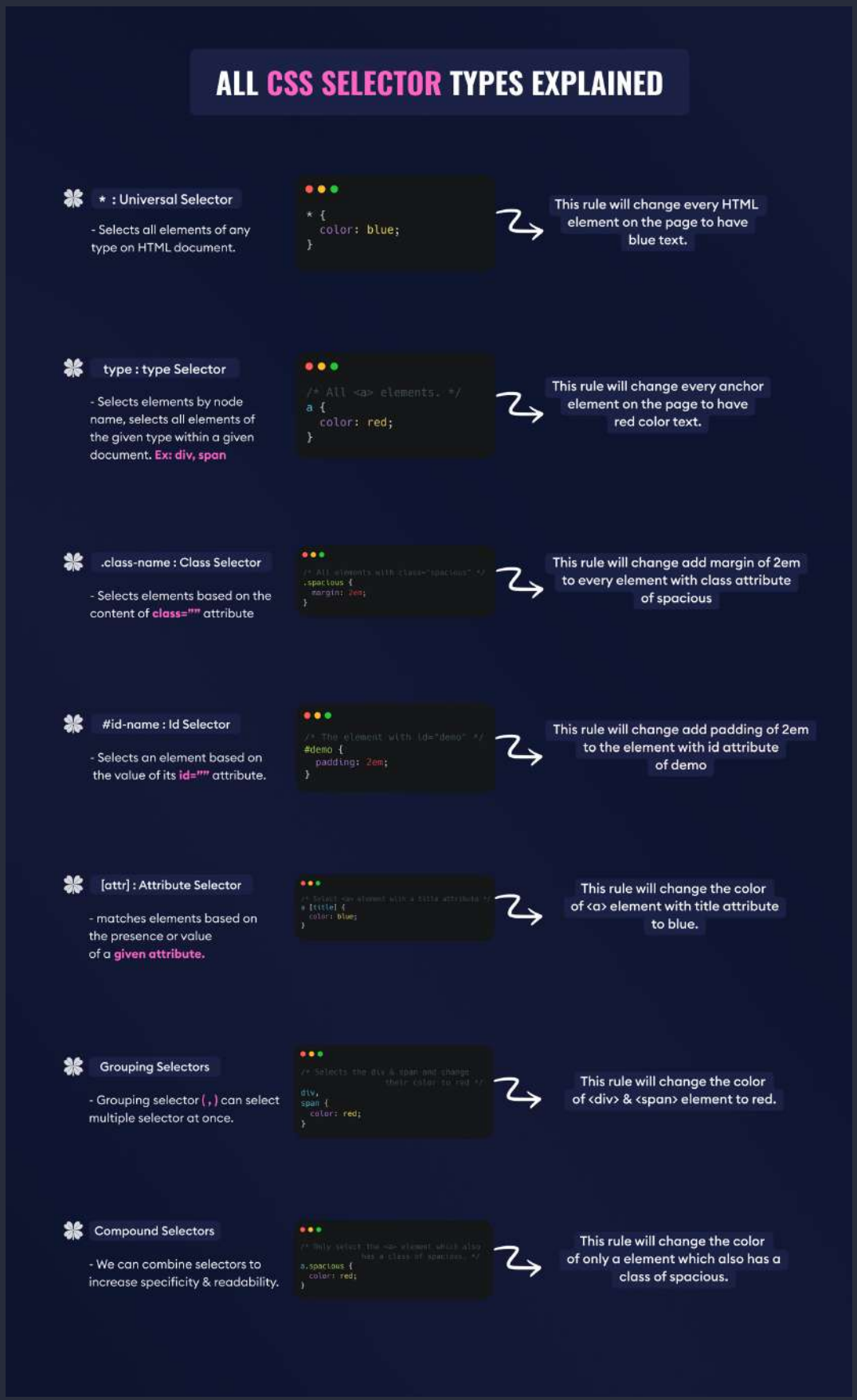
- conic-gradient()
- linear-gradient()
- radial-gradient()
- repeating-linear-gradient()
- repeating-radial-gradient()
- repeating-conic-gradient()
- cross-fade()
- element()
- paint()

SHAPE FUNCTIONS

- circle()
- ellipse()
- inset()
- polygon()
- path()

CODING.STELLA

All CSS Selector Types Explained



CODING.STELLA

All CSS Combinatots Explained

ALL CSS COMBINATORS

Descendant Combinator

- A Descendant Combinator allows us to target a child element.

Select all child & grandchilds of a parent.

```
/* Selects all <strong> elements that are child of p */
p strong {
  color: red;
}
```

This rule will change the color of elements that are child of <p> to blue.

Child Combinator

- Child Combinator helps us to **select direct child of the parent.**

Selects only direct children & not the grandchildren of a parent.

```
/* Selects all <strong> elements that are direct child of p */
p > strong {
  color: blue;
}
```

This rule will change the color of elements that are direct child of <p> to blue.

General sibling Combinator

- A general sibling works in a way that even if the **sibling before the child doesn't the same it will still select it.**

```
/* Selects <strong> elements even if there is something different before last <strong> of <p> it will still select it */
p ~ strong {
  color: red;
}
```

This rule will change all the elements even if the element before the child is not same.

Adjacent Sibling Combinator

- Adjacent sibling works in a way that it will **only select a child matches the first element before it.**

```
/* Selects <strong> elements only if it matches the first element before it */
p + strong {
  color: blue;
}
```

This rule will change the color of elements only if matches the first element before it of <p>

PSEUDO CLASSES & ELEMENTS SELECTOR:-

: Pseudo Classes

- The **:** pseudo allow the selection of elements based on **state information** that not contained in the **document tree.**

Ex: **a:visited**

```
/* Any button over which the user's pointer is hovering will change its background to blue */
button:hover {
  color: blue;
}
```

This rule will change the color of <button> element to blue when user pointer is hovered

:: Pseudo Elements

- The **::** pseudo represent entities that are not included in HTML

Ex: **p::first-line**

```
/* The first line of every <p> element. */
p::first-line {
  color: blue;
}
```

This rule will change the color of every first line of p element to blue

CODING.STELLA

click here!