Contents

[Footer always in bottom 2](#_Toc26306148)

[meaning of `auto` value in a CSS property. 5](#_Toc26306149)

[CSS Variables 5](#_Toc26306150)

[@font-face 5](#_Toc26306151)

[background-blend-mode 6](#_Toc26306152)

[:focus 6](#_Toc26306153)

[An important note for responsive design 7](#_Toc26306154)

[HTML data-attributes 8](#_Toc26306155)

[all: unset; 9](#_Toc26306156)

[adjacent sibling selectors (+) 9](#_Toc26306157)

[general sibling selector (~) 12](#_Toc26306158)

[Inner Shadow 12](#_Toc26306159)

# Footer always in bottom

The simplest solution is to use min-height on the <html> tag and position the <footer> with position:absolute;

html {  
 position: relative;

min-height: 100%;

}

body {

margin: 0 0 100px;

padding: 25px;

}

footer {  
 position:absolute;  
 bottom:0;  
 left:0;  
 width:100%

}

flexbox method

html{ height:100%;}

body{ height:100%; margin:0; }

header{ height:50px; background:lightcyan; }

footer{ height:50px; background:PapayaWhip; }

/\* Trick \*/

body{

display:flex;

flex-direction:column;

}

footer{

margin-top:auto;

}

<body>

<header>Header</header>

<article>Content</article>

<footer>Footer</footer>

</body>

**--------------|or|---------------**

body{ height:100vh; margin:0; }

header{ height:50px; background:lightcyan; }

footer{ height:50px; background:PapayaWhip; }

/\* Trick \*/

body{

display:flex;

flex-direction:column;

}

footer{

margin-top:auto;

}

<body>

<header>Header</header>

<article>Content</article>

<footer>Footer</footer>

</body>

# [meaning of `auto` value in a CSS property.](https://stackoverflow.com/questions/4471850/what-is-the-meaning-of-auto-value-in-a-css-property)

The value of said property is adjusted **automatically** according to the content or the context of the element.

For example, a block-level element with height: auto will grow taller as it contains more text. For another example, a block element with margin: 0 auto will have the left and right margins increased until it becomes centered along the y-axis of the viewport.

It really depends on the property you give the value to, different properties behave differently depending on the content and context.

# CSS Variables

<https://www.youtube.com/watch?v=PHO6TBq_auI>

# @font-face

specify a font named ‘betterfly’, and specify the URL where it can be found:

@font-face{

    font-family: 'betterfly';

    src: url(../font/betterfly.otf);

    font-weight: 400;

}

# background-blend-mode

The background-blend-mode property defines the blending mode of each background layer (color and/or image).

background-blend-mode:normal|multiply|screen|overlay|darken|lighten|color-dodge|saturation|color|luminosity;

<!DOCTYPE html>

<html lang="en">

<head>

    <style>

    .container{

        width: 100wh;

        height: 100vh;

        background-color: lightblue;

**blending (mixing) these two colors**

        background-image: url(thumbnail.jpg);

        background-repeat: no-repeat;

        background-size: cover;

        background-blend-mode:overlay;

    }

    </style>

</head>

<body>

    <div class="container">

    </div>

</body>

</html>

# :focus

.btn-primary:hover,

.btn-primary:focus{ #.btn-primary:focus ; in case if user use keyboard to ...

    background-color: var(--clr-primary-dark);

}

# An important note for responsive design

the naming of elements inside and outside media query should be the same so that it takes effect.

/\* naming is same inside and outside of media queries \*/

section.intro{ padding: 6em;} /\* outside media query \*/

@media only screen and (min-width:786px){

    section.intro{ padding: 6em;} /\* inside media query \*/

}

attr()

<html lang="en">

<head>

    <style>

    .rockets:hover::before {

        content: attr(data-altitude);

    }

    </style>

</head>

<body>

    <div class="rockets" data-speed='4' data-altitude='200' data-distance='700'></div>

</body>

</html>

# HTML data-attributes

Custom data attributes allow authors to create custom data to apply to their HTML elements when no other attributes make sense to use to store extra data.

They help extend the information that html elements can communicate to a program or script. Custom data attributes can be accessed through both JavaScript and CSS.

<html lang="en">

<head>

<style>

    .rockets{

        width: 150px; height: 150px;

        background: url(../rocket.png) no-repeat;

        background-size: 150px;

        transform: rotate(90deg);

        position: absolute; left: 0; transition: 0s linear;

    }

</style>

</head>

<body>

<div class="rockets" data-speed='4' data-altitude='200' data-distance='700'></div>

<div class="rockets" data-speed='5' data-altitude='500' data-distance='700'></div>

<script>

function activateRocket(){

    rockets = document.getElementsByClassName('rockets');

    for (var i = 0; i < rockets.length; i++){

        altitude = rockets[i].dataset.altitude;

        speed = rockets[i].dataset.speed;

        distance = rockets[i].dataset.distance;

        rockets[i].style.left=distance+'px';

        rockets[i].style.top=altitude+'px';

        rockets[i].style.transitionDuration=speed+'s';

    }

}

window.addEventListener('load', activateRocket)

</script>

</body>

</html>

# all: unset;

nav{

    position: absolute;

    top: 100%;

    background-color: var(--bg-color);

    width: 100%;

    left: 0px;

    transform: scale(1, 0);

    margin-bottom: 1em;

    transform-origin: top;

    transition: transform 250ms ease-in-out;

}

@media only screen and (min-width:786px){

    nav{

        all: unset;

    }

}

# adjacent sibling selectors (+)

<https://www.w3schools.com/css/css_combinators.asp>

The adjacent sibling selector selects only one element that’s the adjacent sibling of a specified element.

<!DOCTYPE html>

<html lang="en">

<head>

    <style>

    div + p {

        background-color: yellow;

    }

    </style>

</head>

<body>

    <div>

        <p>paragraph 1</p>

        <p>paragraph 2</p>

    </div>

it selected only one tag, <p>, which is adjacent sibling of <div>

    <p>Paragraph 3</p>

    <p>Paragraph 4</p>

    <p>Paragraph 5</p>

</body>

</html>

**-----------------------------**

<!DOCTYPE html>

<html lang="en">

<head>

    <style>

    p + p {

        background-color: yellow;

    }

    </style>

</head>

<body>

    <div>

        <p>paragraph 1</p>

        <p>paragraph 2</p>

        <p>paragraph 3</p>

    </div>

    <p>Paragraph 3</p>

it selected only one tag, <p>, which is adjacent sibling of this paragraph tag, <p>, but since there is a lot of p tag followed by one another it selects one <p> tag that comes after one another and that is why it highlighted lots of those p tags.

    <p>Paragraph 4</p>

    <p>Paragraph 5</p>

    <p>Paragraph 6</p>

</body>

</html>

# general sibling selector (~)

The general sibling selector selects all elements that are siblings of a specified element.

<!DOCTYPE html>

<html lang="en">

<head>

    <style>

    div ~ p {

        background-color: yellow;

    }

    </style>

</head>

<body>

    <div>

        <p>paragraph 1</p>

        <p>paragraph 2</p>

    </div>

    <p>Paragraph 3</p>

    <p>Paragraph 4</p>

    <code>This is a code.</code>

    <p>Paragraph 5</p>

</body

</html>

# Inner Shadow

<https://www.youtube.com/watch?v=P7t13SGytRk>

box-shadow: rgb(0, 0, 0) 120px 100px 250px inset,

rgb(0, 0, 0) -120px -100px 250px inset;

Bottom Shadow

Right Shadow

Top Shadow

Left Shadow