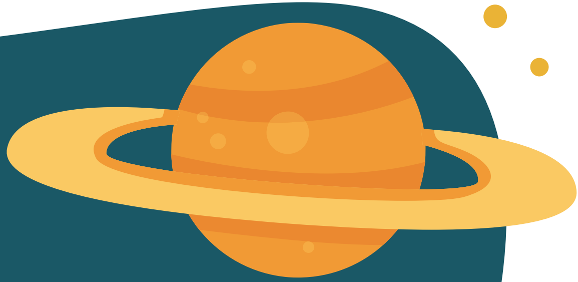
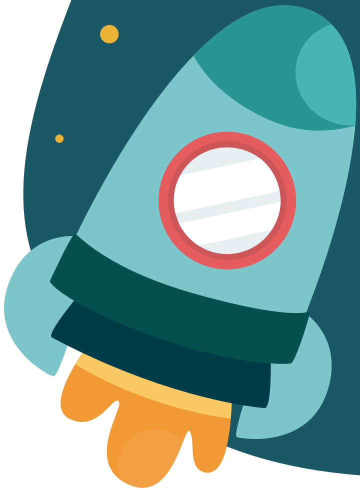


CSS

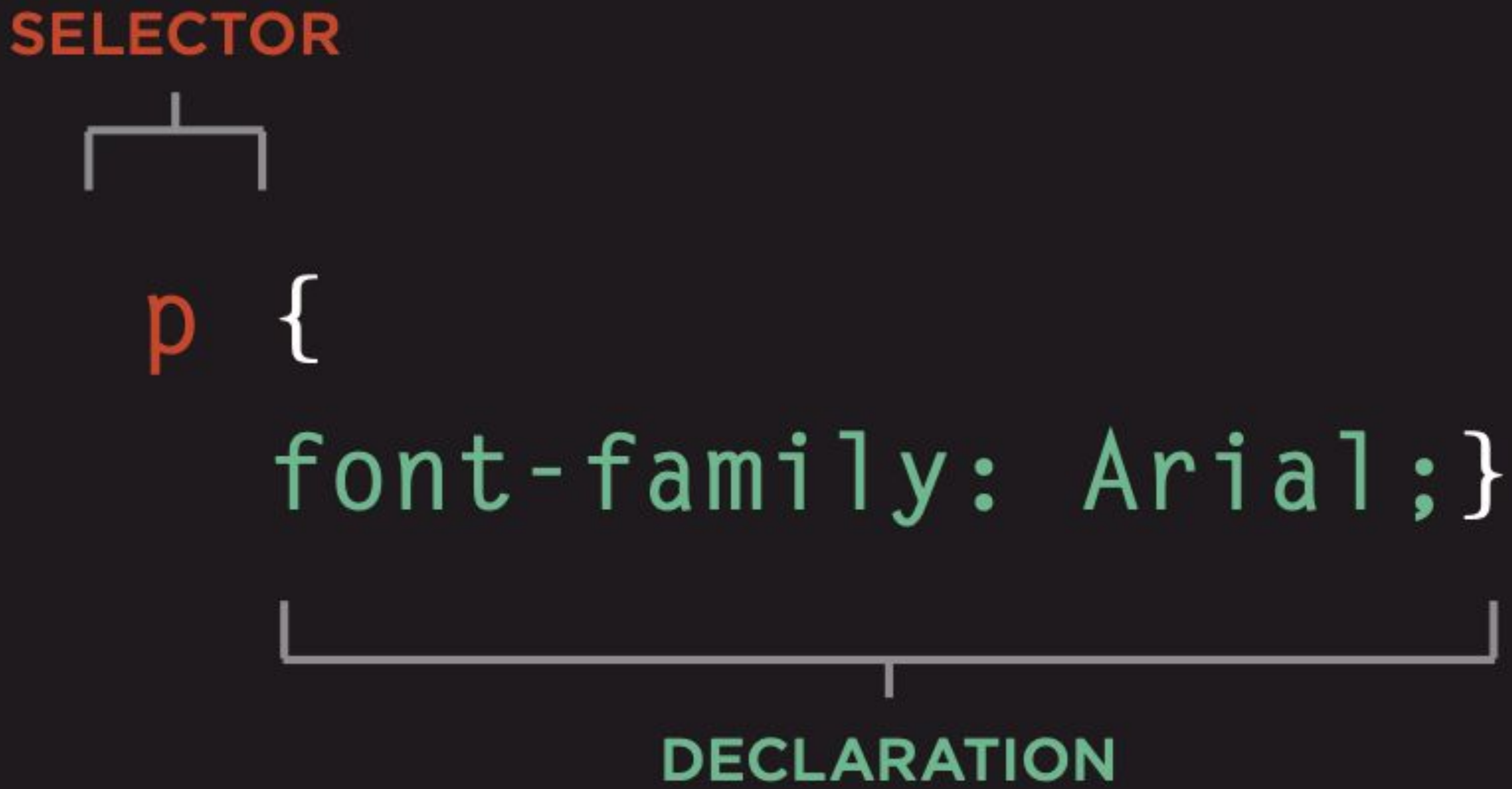
Cascading Style Sheets

88823764

Software Development Training Camp



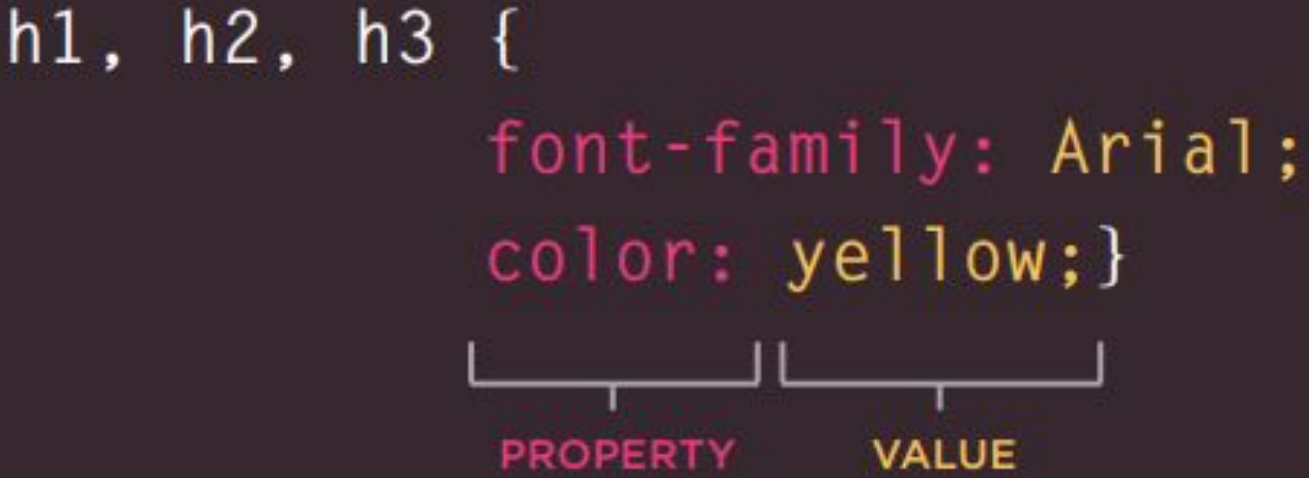
CSS Basic



SELECTOR

```
p {
font-family: Arial;}
```

DECLARATION



```
h1, h2, h3 {
           font-family: Arial;
           color: yellow;}
```

PROPERTY

VALUE

CSS

- There are three ways of inserting a style sheet:
    - External CSS
    - Internal CSS
    - Inline CSS

External CSS

```
<!DOCTYPE html>
<html>
    <head>
        <link rel=″stylesheet″ href=″mystyle.css″>
    </head>
    <body>
        <h1>This is a heading</h1>
        <p>This is a paragraph.</p>
    </body>
</html>
```

Internal CSS

```
<style>
    body {
      background-color: linen;
    }

    h1 {
      color: maroon;
      margin-left: 40px;
    }
</style>
```

Inline CSS

```
<!DOCTYPE html>
<html>
    <body>

    <h1 style="color:blue;text-align:center;">This is a heading</h1>
    <p style="color:red;">This is a paragraph.</p>

    </body>
</html>
```

CSS Selectors

```
p { /* tag */
  text-align: center;
  color: red;
}

#para1 { /* #id */
  text-align: center;
  color: red;
}

.center { /* .class */
  text-align: center;
  color: red;
}
```

```
* { /* Universal Selector */
  text-align: center;
  color: blue;
}
```

CSS Units

- CSS มีการกำหนดค่าความยาว "length" ของ property
  - width
  - margin
  - padding
  - font-size
  - etc.
- Absolute Lengths
- Relative Lengths

Absolute Lengths – a length fixed

Unit	Description
cm	centimeters
mm	millimeters
in	inches (1in = 96px = 2.54cm)
px	pixels (1px = 1/96th of 1in)
pt	points (1pt = 1/72 of 1in)
pc	picas (1pc = 12 pt)

Relative Lengths – a length relative to another length property

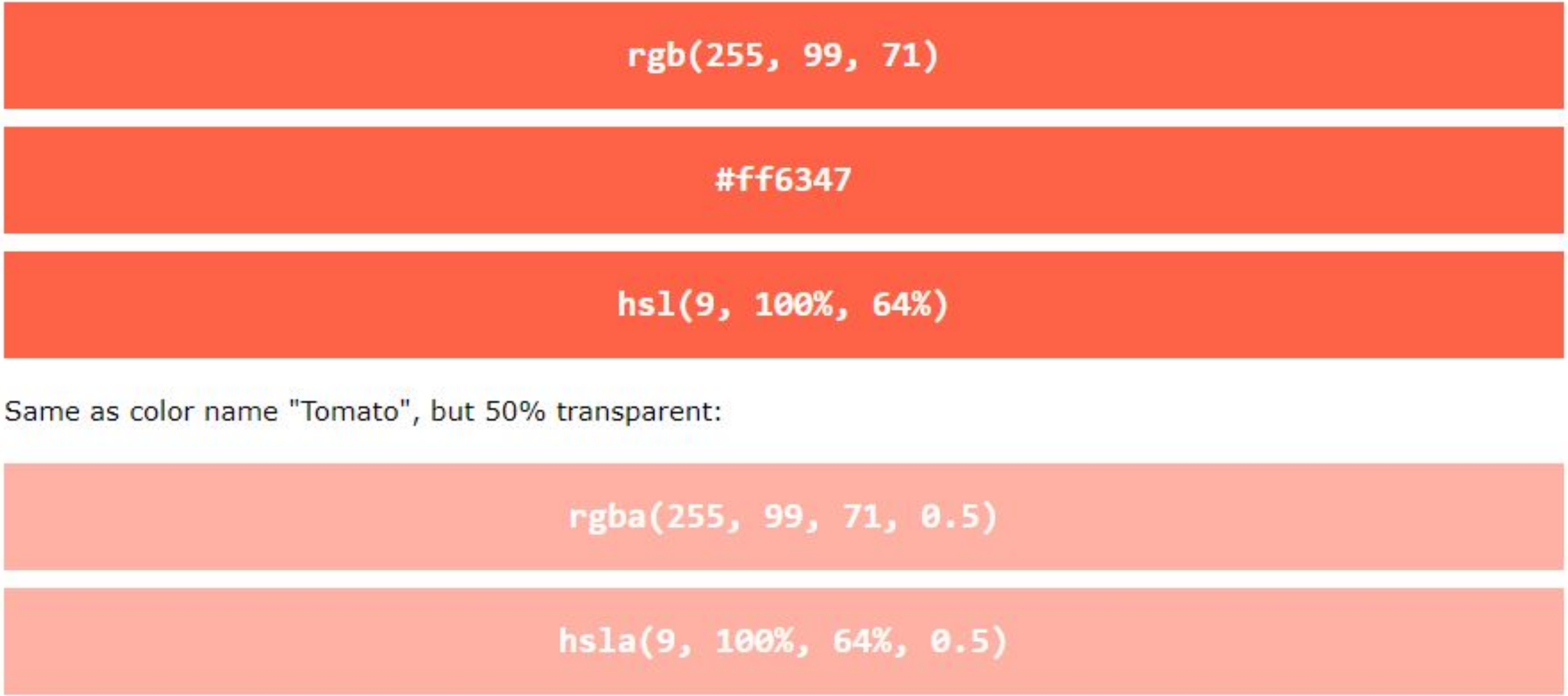
Unit	Description
em	Relative to the font-size of the element (2em means 2 times the size of the current font)
ex	Relative to the x-height of the current font (rarely used)
ch	Relative to the width of the "0" (zero)
rem	Relative to font-size of the root element
vw	Relative to 1% of the width of the viewport*
vh	Relative to 1% of the height of the viewport*
vmin	Relative to 1% of viewport's* smaller dimension
vmax	Relative to 1% of viewport's* larger dimension
%	Relative to the parent element

CSS Colors

```
<h1 style="background-color:DodgerBlue;">Hello World</h1>

<h1 style="color:Tomato;">Hello World</h1>
```

CSS Color Values



rgb(255, 99, 71)

#ff6347

hsl(9, 100%, 64%)

Same as color name "Tomato", but 50% transparent:

rgba(255, 99, 71, 0.5)

hsla(9, 100%, 64%, 0.5)

CSS Fonts: font-family

```
.p1 {
  font-family: "Times New Roman", Times, serif;
}

.p2 {
  font-family: Arial, Helvetica, sans-serif;
}

.p3 {
  font-family: "Lucida Console", "Courier New", monospace;
}
```

CSS Fonts: @font-face

```
@font-face {
  font-family: myFirstFont;
  src: url(sansation_light.woff);
}

div {
  font-family: myFirstFont;
}
```

CSS Fonts: font-style

```
p.normal {
  font-style: normal;
}

p.italic {
  font-style: italic;
}
```

CSS Fonts: font-weight

```
p.normal {
  font-style: normal;
  font-weight: bold;
}

p.italic {
  font-style: italic;
  font-weight: bold;
}
```

CSS Fonts: font-size (1/2)

```
h1 {
  font-size: 40px;
}

h2 {
  font-size: 30px;
}

p {
  font-size: 14px;
}
```

CSS Fonts: font-size (2/2)

```
h1 {
  font-size: 2.5em; /* 40px/16=2.5em */
}

h2 {
  font-size: 1.875em; /* 30px/16=1.875em */
}

p {
  font-size: 0.875em; /* 14px/16=0.875em */
}
/* The default text size in browsers is 16px */
```

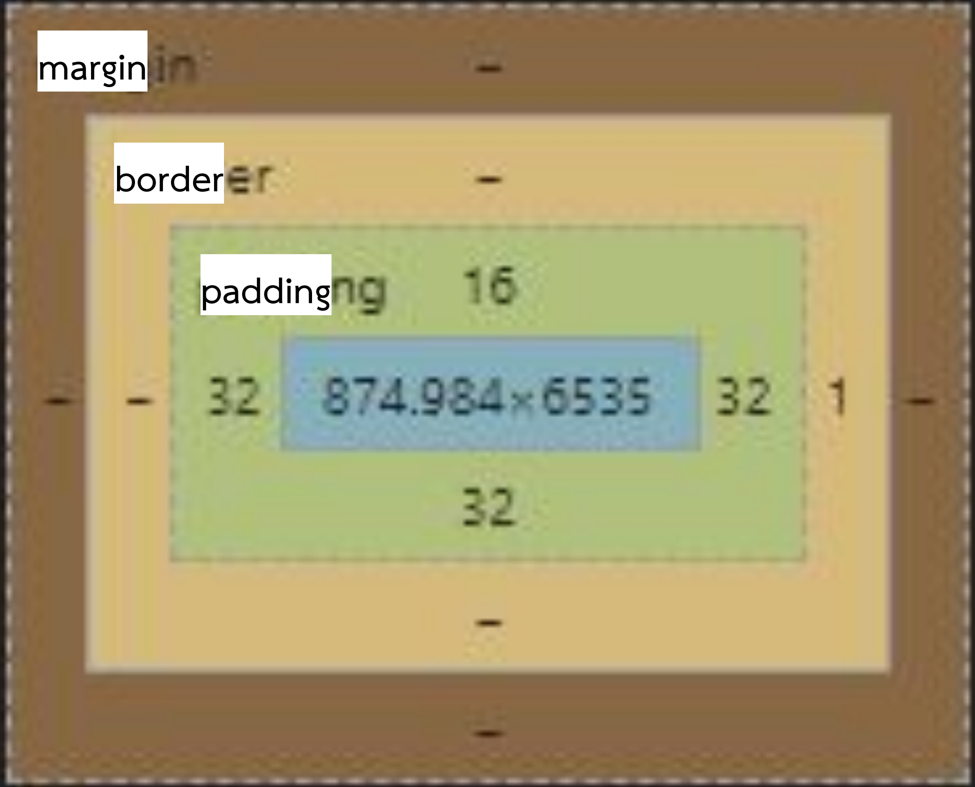
CSS Fonts: import (1/2)

```
<head>
    <link rel="stylesheet"
    href="https://fonts.googleapis.com/css?family=Sofia">
    <style>
        body {
          font-family: "Sofia", sans-serif;
        }
    </style>
</head>
```

CSS Fonts: import (2/2)

```
<head>
        <style>
                @import url(https://fonts.googleapis.com/css?family=Sofia);
                body{
                   font-family: 'Sofia',serif;
                }
        </style>
</head>
```

CSS Element Property


margin
–
border
–
padding
16
–
–
32
874.984×6535
32
1
–
32
–
–

CSS Border

- CSS Border - Shorthand Property
  - border-width
  - border-style (required)
  - border-color

```
p {
 border: 5px solid red;
}

h1{
   border-style: dashed;
   border-width:  2px;
   border-radius: 5px;
   border-top-style: dotted;
}
```

CSS Margin

- Margins are used to create space around elements, outside of any defined borders.

```
p {
  margin-top: 100px;
  margin-bottom: 100px;
  margin-right: 150px;
  margin-left: 80px;
}
```

- Margin - Shorthand Property
    - margin: 25px 50px 75px 100px;
        - top margin is 25px
        - right margin is 50px
        - bottom margin is 75px
        - left margin is 100px

CSS Padding

- Padding is used to create space around an element's content, inside of any defined borders.

```
p {
  padding-top: 50px;
  padding-right: 30px;
  padding-bottom: 50px;
  padding-left: 80px;
}
```

- Padding - Shorthand Property
  - padding: 25px 50px 75px 100px;
    - top padding is 25px
    - right padding is 50px
    - bottom padding is 75px
    - left padding is 100px

CSS Pseudo-classes

```
/* unvisited link */
a:link {
  color: #FF0000;
}

/* visited link */
a:visited {
  color: #00FF00;
}
```

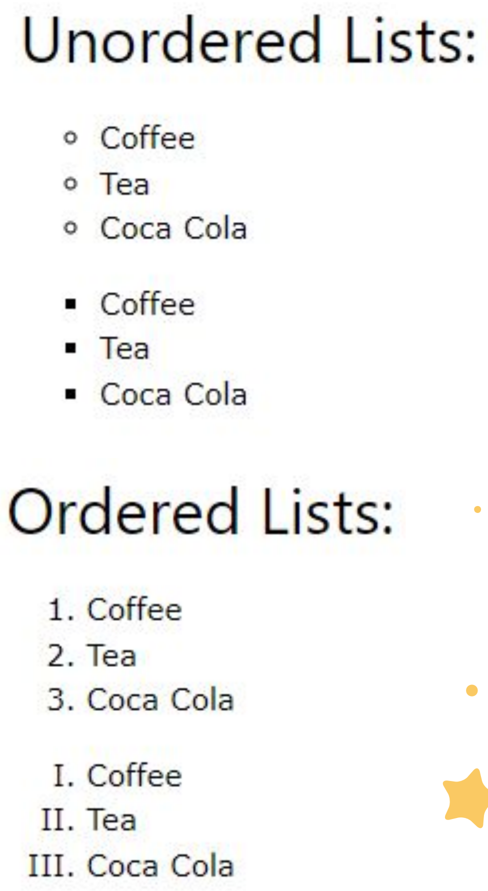
```
/* mouse over link */
a:hover {
  color: #FF00FF;
}

/* selected link */
a:active {
  color: #0000FF;
}
```

CSS Lists

```
ul.a {
  list-style-type: circle;
}
ul.b {
  list-style-type: square;
}

ol.c {
  list-style-type: upper-roman;
}
ol.d {
  list-style-type: lower-alpha;
}
```



Unordered Lists:

○ Coffee
○ Tea
○ Coca Cola

■ Coffee
■ Tea
■ Coca Cola

Ordered Lists:

1. Coffee
2. Tea
3. Coca Cola

I. Coffee
II. Tea
III. Coca Cola

CSS List – Shorthand property

```
ul {
  list-style: square inside url(”sqpurple.gif”);
}
```

- list-style-type
- list-style-position
- list-style-image

CSS Layout

CSS Layout

display

CSS Layout – The display Property

- property specifies if/how an element is displayed

- block
- inline
- none

```
li {
  display: inline;
}
```

Examples of block-level elements

<div>

<h1> - <h6>

<p>

<form>

<header>

<footer>

<section>

<ul> - <ol>

Examples of Inline-level elements

<span>

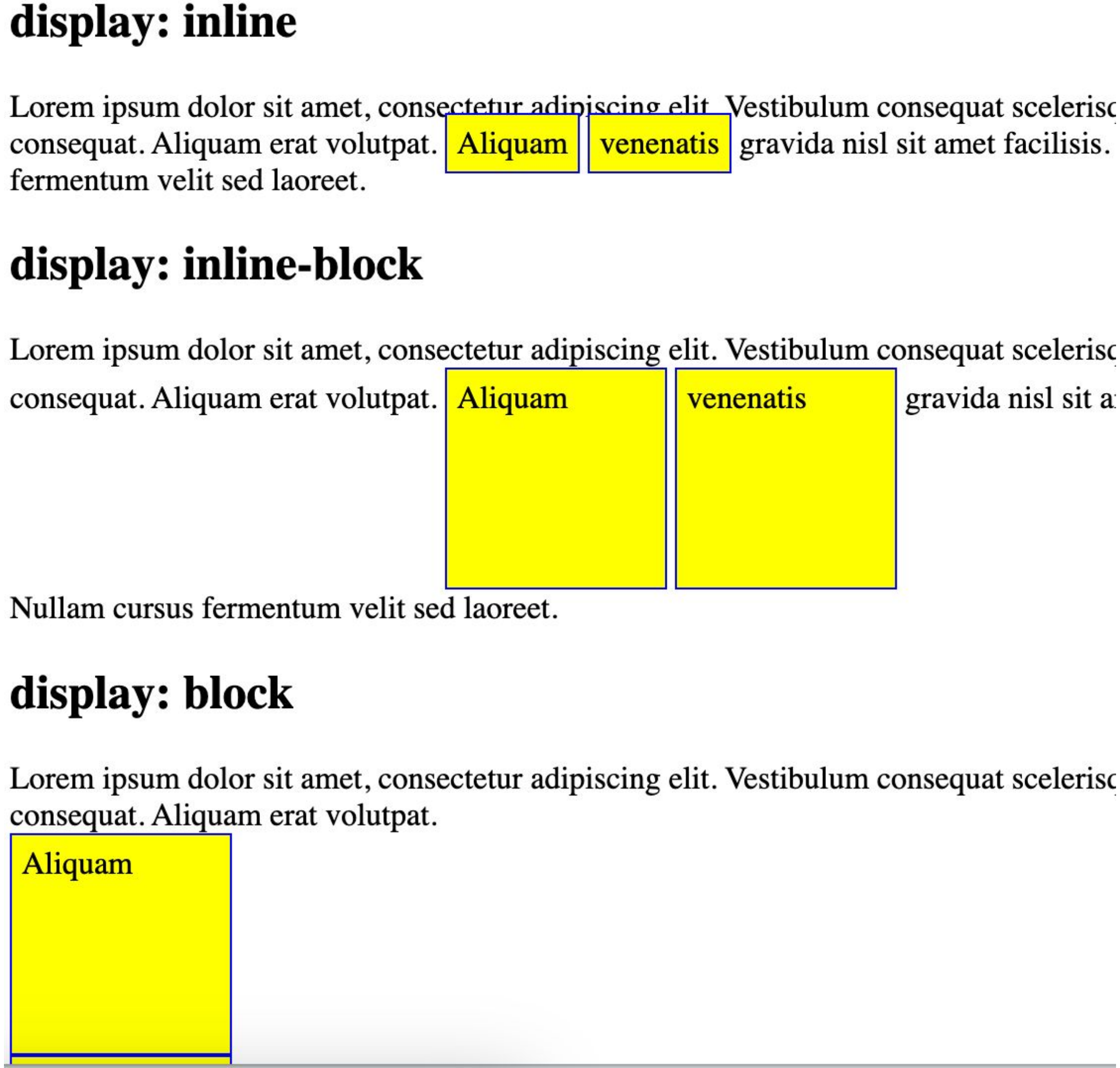
<a>

<img>

Hide an Element

```
h1.hidden {
  display: none;
}
```

```
h1.hidden {
  visibility: hidden;
}
```



display: inline

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisc consequat. Aliquam erat volutpat. Aliquam venenatis gravida nisl sit amet facilisis. fermentum velit sed laoreet.

display: inline-block

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisc consequat. Aliquam erat volutpat. Aliquam venenatis gravida nisl sit a

Nullam cursus fermentum velit sed laoreet.

display: block

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisc consequat. Aliquam erat volutpat.

Aliquam

CSS Layout

position

CSS Layout – The position Property

- คุณสมบัติระบุประเภทของวิธีการวางตำแหน่งที่ใช้สำหรับองค์ประกอบ
- Position values
  - static
  - relative
  - fixed
  - absolute
  - sticky

position: static;

- elements are positioned static by default.

```
div.static {
  position: static;
  border: 3px solid #73AD21;
}
```

position: relative;

- properties of a relatively-positioned element will be adjusted away from its normal position.

```
div.relative {
  position: relative;
  left: 30px;
  border: 3px solid #73AD21;
}
```

position: fixed;

- fixed element does not leave a gap in the page.

```
div.fixed {
  position: fixed;
  bottom: 0;
  right: 0;
  width: 300px;
  border: 3px solid #73AD21;
}
```

position: sticky;

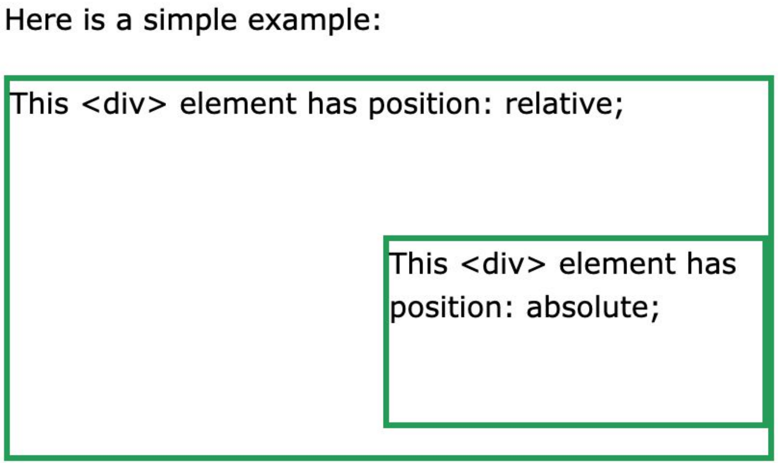
- sticky element toggles between relative and fixed
- depending on the scroll position

```
div.sticky {
  position: -webkit-sticky; /* Safari */
  position: sticky;
  top: 0;
  background-color: green;
  border: 2px solid #4CAF50;
}
```

position: absolute;

- Absolute positioned elements are removed from the normal flow, and can **overlap** elements.

```
div.absolute {
  position: absolute;
  top: 80px;
  right: 0;
  width: 200px;
  height: 100px;
  border: 3px solid #73AD21;
}
```



Here is a simple example:

This <div> element has position: relative;

This <div> element has position: absolute;

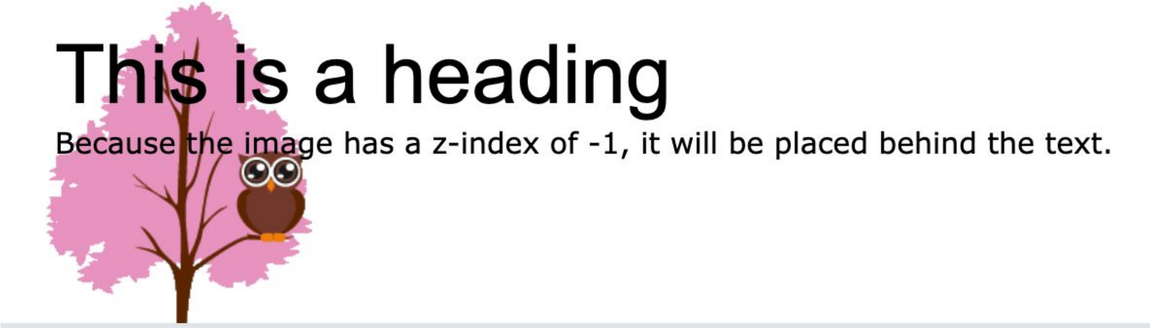
CSS Layout

z-index

CSS Layout – The z-index Property

- stack order of an element

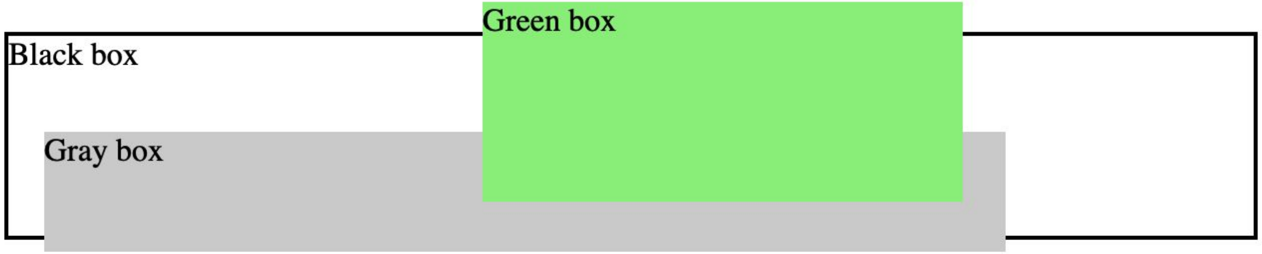
```
img {
  position: absolute;
  left: 0px;
  top: 0px;
  z-index: -1;
}
```



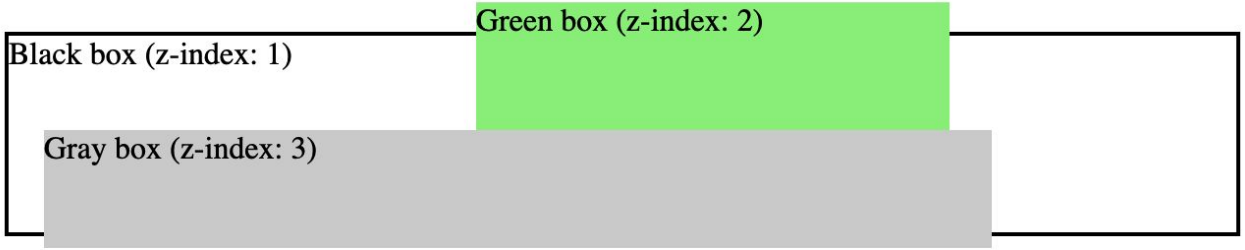
This is a heading

Because the image has a z-index of -1, it will be placed behind the text.

- With out z-index



- With z-index



CSS Layout

Overflow

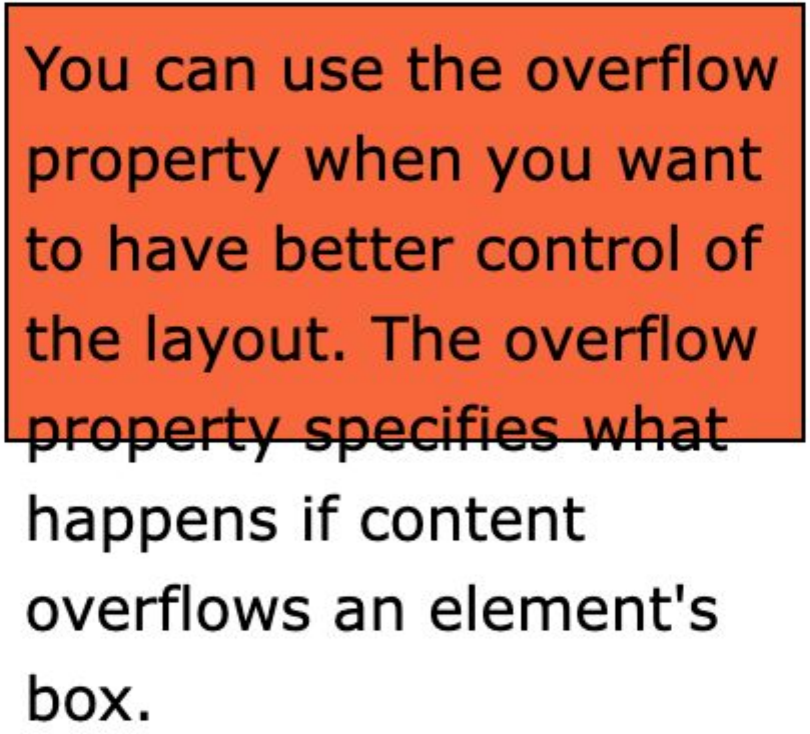
CSS Layout – Overflow

- add scrollbars when the content of an element is too big to fit in the specified area

- values
  - visible
  - hidden
  - scroll
  - auto

overflow: visible;

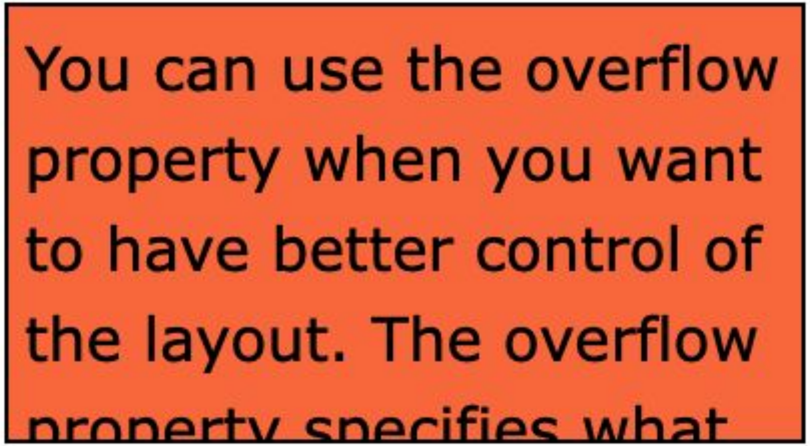
```
div {
  width: 200px;
  height: 65px;
  background-color: coral;
  overflow: visible;
}
```



You can use the overflow property when you want to have better control of the layout. The overflow property specifies what happens if content overflows an element's box.

overflow: hidden;

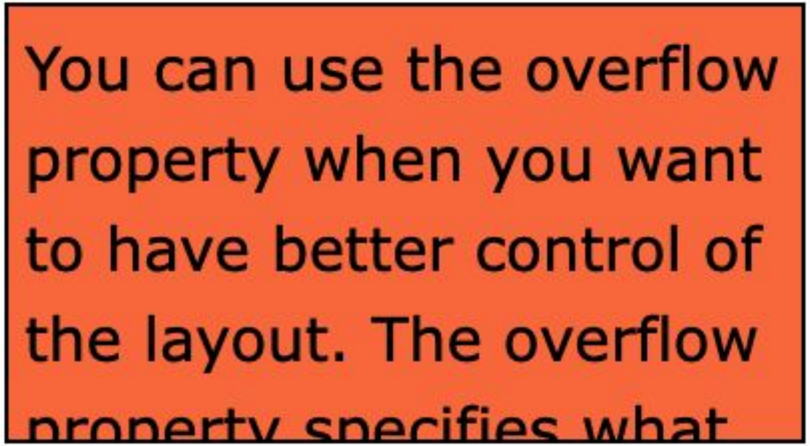
```
div {
  width: 200px;
  height: 65px;
  background-color: coral;
  overflow: hidden;
}
```



You can use the overflow property when you want to have better control of the layout. The overflow property specifies what

overflow: scroll;

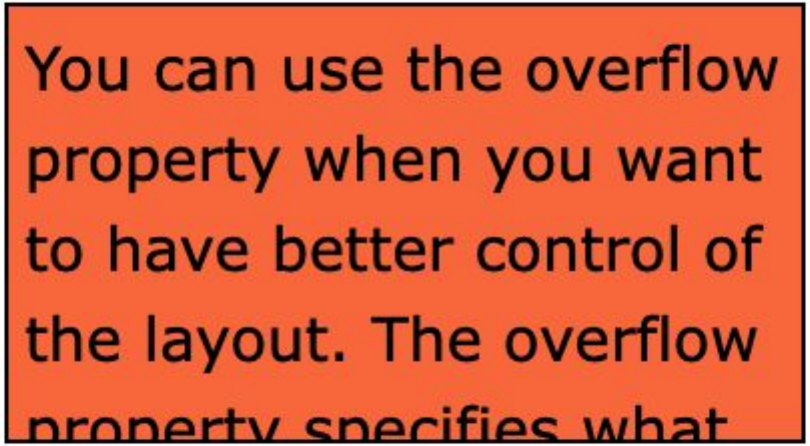
```
div {
  width: 200px;
  height: 65px;
  background-color: coral;
  overflow: scroll;
}
```



You can use the overflow property when you want to have better control of the layout. The overflow property specifies what

overflow: auto;

```
div {
  width: 200px;
  height: 65px;
  background-color: coral;
  overflow: scroll;
}
```



You can use the overflow property when you want to have better control of the layout. The overflow property specifies what

overflow-x and overflow-y

- **overflow-x** specifies what to do with the **left/right** edges of the content.

- **overflow-y** specifies what to do with the **top/bottom** edges of the content.

```
div {
  overflow-x: hidden; /* Hide horizontal scrollbar */
  overflow-y: scroll; /* Add vertical scrollbar */
}
```

CSS Layout

float and clear

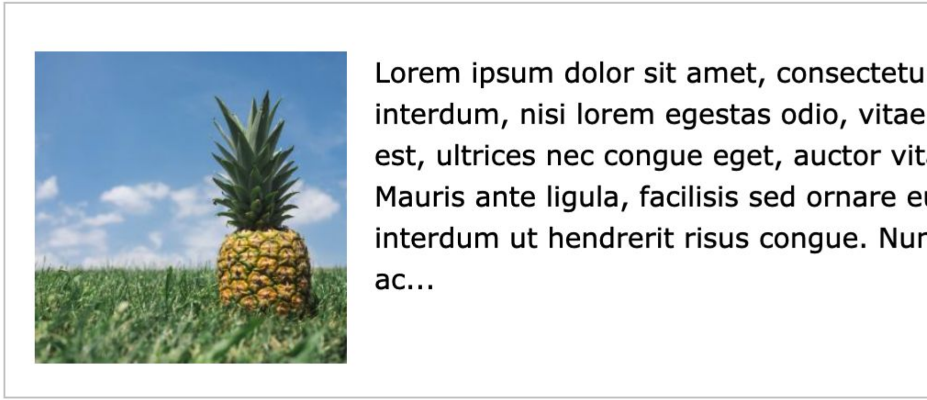
CSS Layout – float and clear

- float
  - ระบุว่าองค์ประกอบควรลอยอย่างไร
- clear
  - ระบุเพื่อให้องค์ประกอบข้างๆ float หรือก่อน clear มีการนับ float ใหม่

CSS Layout – float

- values
  - left
  - right
  - none
  - inherit

```
img {
  float: left;
}
```



Lorem ipsum dolor sit amet, consectetu
interdum, nisi lorem egestas odio, vitae
est, ultrices nec congue eget, auctor vit
Mauris ante ligula, facilisis sed ornare e
interdum ut hendrerit risus congue. Nur
ac...

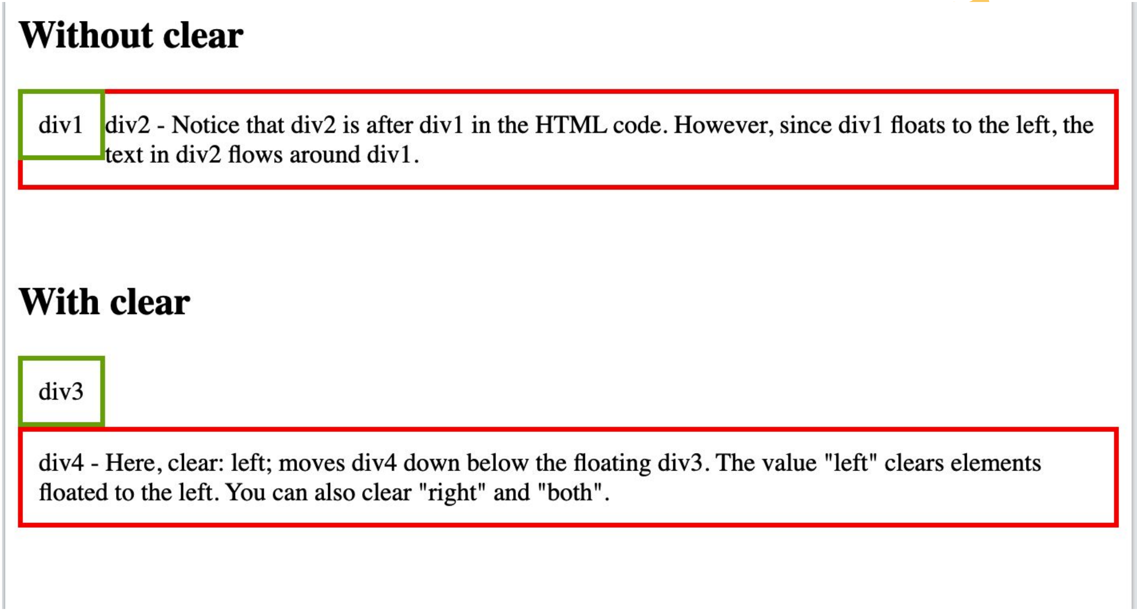
CSS Layout – clear

- values
  - none
  - left
  - right
  - both
  - inherit

CSS Layout – clear

```
.div3 {
  float: left;
  padding: 10px;
  border: 3px solid #73AD21;
}

.div4 {
  padding: 10px;
  border: 3px solid red;
  clear: left;
}
```



Without clear

div1

div2 - Notice that div2 is after div1 in the HTML code. However, since div1 floats to the left, the text in div2 flows around div1.

With clear

div3

div4 - Here, clear: left; moves div4 down below the floating div3. The value "left" clears elements floated to the left. You can also clear "right" and "both".

CSS Layout

Horizontal & Vertical Align

CSS Layout – Horizontal & Vertical Align

```
.center {
  margin: auto;
  width: 50%;
  border: 3px solid green;
  padding: 10px;
}

.center {
  text-align: center;
  border: 3px solid green;
}
```

```
img {
  display: block;
  margin-left: auto;
  margin-right: auto;
  width: 40%;
}
```

CSS Combinators

CSS Combinators

- descendant selector (space)
- child selector (>)
- adjacent sibling selector (+)
- general sibling selector (~)

CSS Combinators: descendant selector (space)

```
div  p {
  background-color: yellow;
}
```

```
<div>
  <p>Paragraph 1 in the div.</p>
  <section>
    <p>Paragraph 2 in the div.</p>
  </section>
</div>
<p>Paragraph 3. After a div.</p>
<p>Paragraph 4. After a div.</p>
<div>
  <p>Paragraph 5 in the div.</p>
</div>
```

CSS Combinators: child selector (>)

```
div > p {
  background-color: yellow;
}
```

```
<div>
  <p>Paragraph 1 in the div.</p>
  <section>
    <p>Paragraph 2 in the div.</p>
  </section>
</div>
<p>Paragraph 3. After a div.</p>
<p>Paragraph 4. After a div.</p>
<div>
  <p>Paragraph 5 in the div.</p>
</div>
```

CSS Combinators: adjacent sibling selector (+)

```css
div + p {
  background-color: yellow;
}
```

```html
<div>
  <p>Paragraph 1 in the div.</p>
  <section>
    <p>Paragraph 2 in the div.</p>
  </section>
</div>
<p>Paragraph 3. After a div.</p>
<p>Paragraph 4. After a div.</p>
<div>
  <p>Paragraph 5 in the div.</p>
</div>
```

CSS Combinators: general sibling selector (~)

```
div ~ p {
  background-color: yellow;
}
```

```
<div>
  <p>Paragraph 1 in the div.</p>
  <section>
    <p>Paragraph 2 in the div.</p>
  </section>
</div>
<p>Paragraph 3. After a div.</p>
<p>Paragraph 4. After a div.</p>
<div>
  <p>Paragraph 5 in the div.</p>
</div>
```

CSS Attribute Selectors

CSS Attribute Selectors

```
a[target] {
  background-color: yellow;
}
```

CSS [attribute="value"] Selector

```
a[target=_blank] {
  background-color: yellow;
}
```

```
<a
href="https://www.w3schools.com">
w3schools.com</a>

<a href="http://www.disney.com"
target="_blank">disney.com</a>

<a href="http://www.wikipedia.org"
target="_top">wikipedia.org</a>
```

CSS [attribute~="value"] Selector

```
[title~=flower] {
  border: 5px solid yellow;
}
```

```
<img src="klematis.jpg"
title="klematis flower" width="150"
height="113">

<img src="img_flwr.gif"
title="flower" width="224"
height="162">

<img src="img_tree.gif" title="tree"
width="200" height="358">
```

CSS [attribute|="value"] Selector

```
[class|="top"] {
  background: yellow;
}
```

```
<h1 class="top-header">Welcome</h1>

<p class="top-text">Hello world!</p>

<p class="topcontent">Are you learning
CSS?</p>
```

CSS [attribute^="value"] Selector

```
[class^="top"] {
  background: yellow;
}
```

```
<h1 class="top-header">Welcome</h1>
<p class="top-text">Hello world!</p>
<p class="topcontent">Are you learning CSS?</p>
```

CSS [attribute$="value"] Selector

```
[class$="test"] {
  background: yellow;
}
```

<div class="first_test">The first div element.</div>

<div class="second">The second div element.</div>

<div class="my-test">The third div element.</div>

<p class="mytest">This is some text in a paragraph.</p>

CSS [attribute*="value"] Selector

```
[class*="te"] {
  background: yellow;
}
```

<div class="first_test">The first div element.</div>

<div class="second">The second div element.</div>

<div class="my-test">The third div element.</div>

<p class="mytest">This is some text in a paragraph.</p>

!important

CSS The !important Rule

- add more importance to a property/value than normal.

```
#myid {
  background-color: blue;
}

.myclass {
  background-color: gray;
}

p {
  background-color: red !important;
}
```

Media Queries

CSS Media Queries

```
@media not|only mediatype and (mediafeature and|or|not mediafeature) {
  CSS-Code;
}

@media only screen and (max-width: 600px) {
  div.example {
    font-size: 30px;
  }
```