Cascading Style Sheet (CSS)

Internet Programming I: Chapter 3 – Part II

ADDIS ABABA SCIENCE AND TECHNOLOGY UNIVERSITY

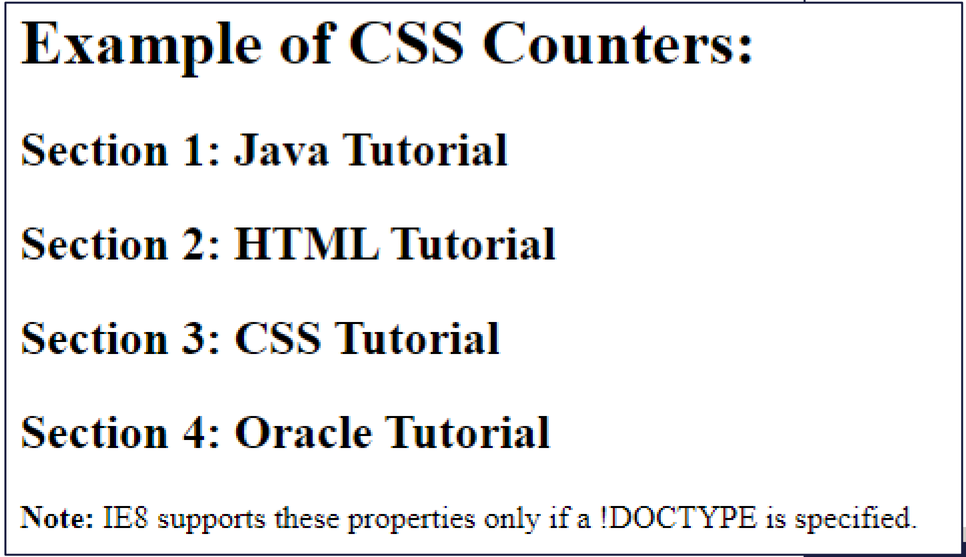
Department of Software Engineering

Main Source: https://www.javatpoint.com/

CSS counters

- **counters** are similar to variables, maintained by CSS and those values can be incremented by CSS rules to track how many times they are used
- CSS Counter Properties
  - **counter-reset**: It is used to create or reset a counter
  - **counter-increment**: It is used to increment the counter value
  - **content**: It is used to insert generated content
  - **counter()** or **counters()** function: It is used to add the value of a counter to an element

```
body {
   counter-reset: section;
}
h2::before {
   counter-increment: section;
   content: "Section " counter(section) ": ";
}
… …
<h1>Example of CSS Counters:</h1>
<h2>Java Tutorial</h2>
<h2>HTML Tutorial</h2>
<h2>CSS Tutorial</h2>
<h2>Oracle Tutorial</h2>
<p><strong>Note:</strong> IE8 supports these properties only if a !DOCTYPE is specified.</p>
```



Example of CSS Counters:

Section 1: Java Tutorial

Section 2: HTML Tutorial

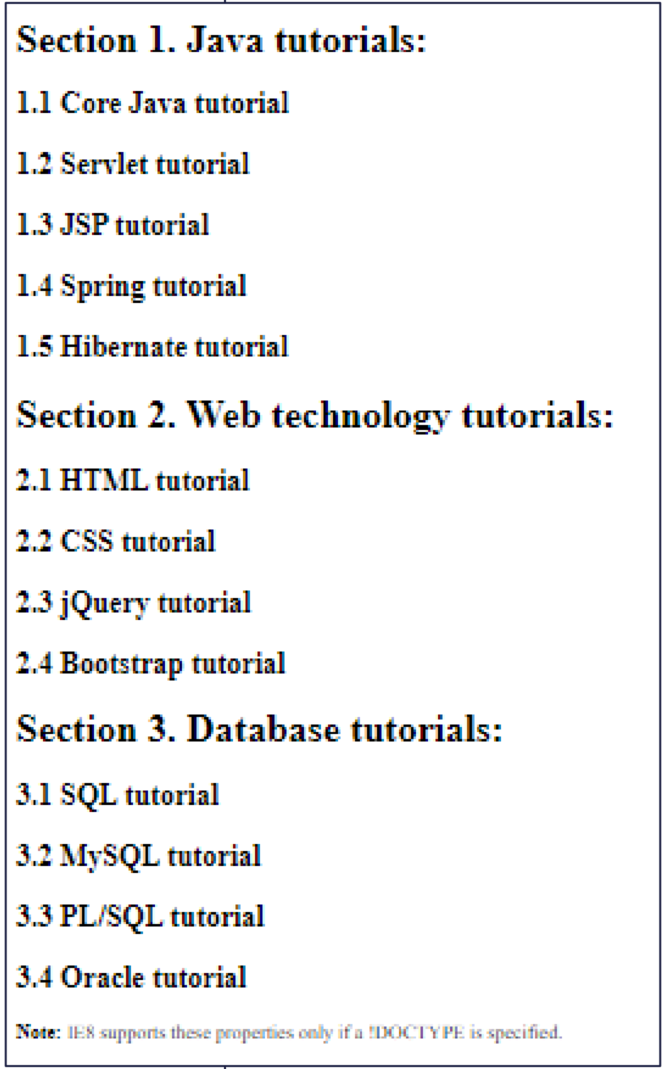
Section 3: CSS Tutorial

Section 4: Oracle Tutorial

Note: IE8 supports these properties only if a !DOCTYPE is specified.

Nesting Counters example

```
body {  counter-reset: section; }
h1 {  counter-reset: subsection;}
h1::before {
   counter-increment: section;
   content: "Section " counter(section) ". "; }
h2::before {
   counter-increment: subsection;
   content: counter(section) "." counter(subsection) " ";  }
… …
<h1>Java tutorials:</h1>
<h2>Core Java tutorial</h2>
<h2>Servlet tutorial</h2>
<h2>JSP tutorial</h2>
<h2>Spring tutorial</h2>
<h2>Hibernate tutorial</h2>

<h1>Web technology tutorials:</h1>
<h2>HTML tutorial</h2>
<h2>CSS tutorial</h2>
<h2>jQuery tutorial</h2>
<h2>Bootstrap tutorial</h2>

<h1>Database tutorials:</h1>
<h2>SQL tutorial</h2>
<h2>MySQL tutorial</h2>
<h2>PL/SQL tutorial</h2>
<h2>Oracle tutorial</h2>
<p><strong>Note:</strong> IE8 supports these properties only if a
!DOCTYPE is specified.</p>
```



Section 1. Java tutorials:

1.1 Core Java tutorial

1.2 Servlet tutorial

1.3 JSP tutorial

1.4 Spring tutorial

1.5 Hibernate tutorial

Section 2. Web technology tutorials:

2.1 HTML tutorial

2.2 CSS tutorial

2.3 jQuery tutorial

2.4 Bootstrap tutorial

Section 3. Database tutorials:

3.1 SQL tutorial

3.2 MySQL tutorial

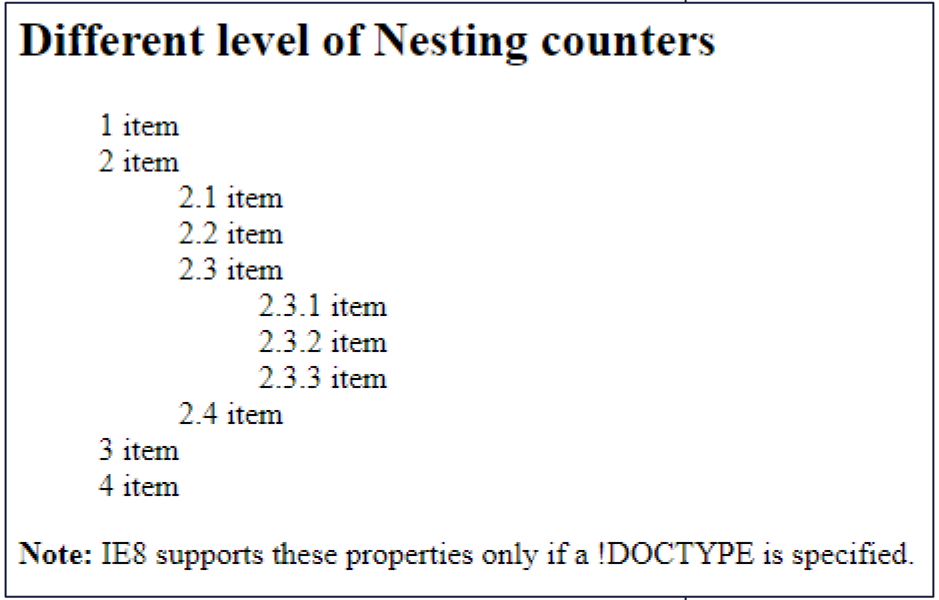
3.3 PL/SQL tutorial

3.4 Oracle tutorial

Note: IE8 supports these properties only if a !DOCTYPE is specified.

Different level of nesting counters example

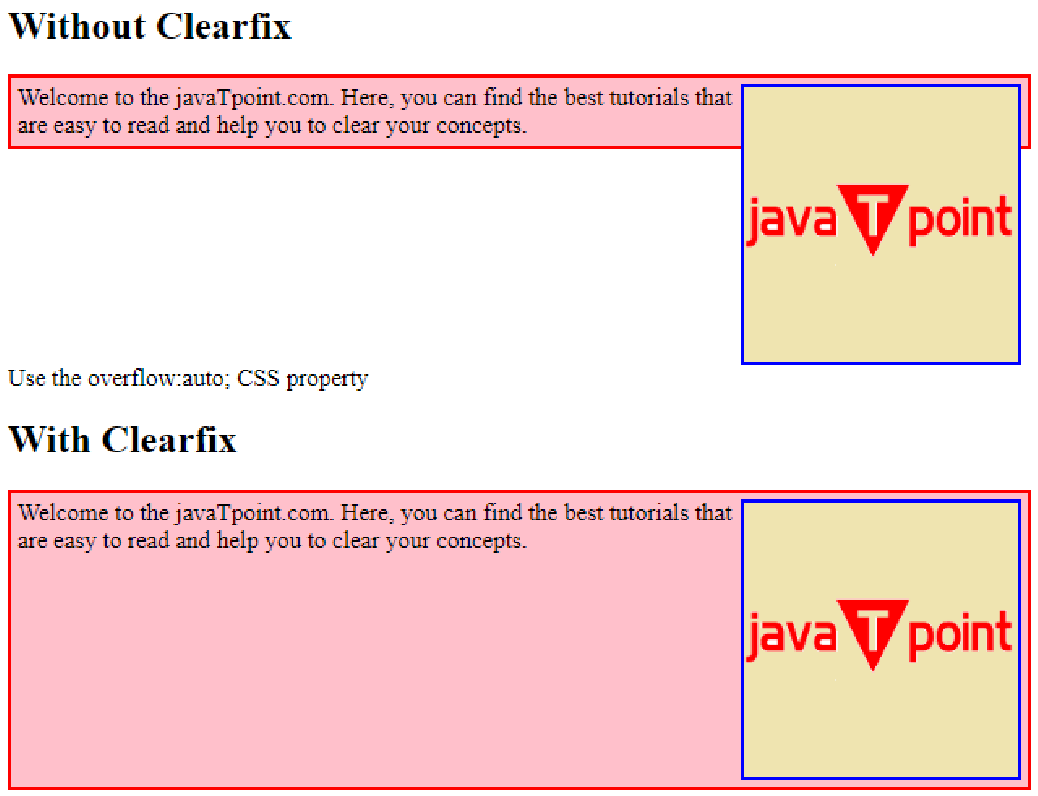
```
ol {  counter-reset: section;
   list-style-type: none; }
li::before {  counter-increment: section;
   content: counters(section,".") " ";  }
… …
<h2>Different level of Nesting counters</h2>
<ol>
  <li>item</li>
  <li>item
    <ol>
      <li>item</li>
      <li>item</li>
      <li>item
        <ol>
          <li>item</li>
          <li>item</li>
          <li>item</li>
        </ol>
      </li>
      <li>item</li>
    </ol>
  </li>
  <li>item</li>
  <li>item</li>
</ol>
<p><b>Note:</b> IE8 supports these properties only if a !DOCTYPE is specified.</p>
```



```
Different level of Nesting counters

    1 item
    2 item
        2.1 item
        2.2 item
        2.3 item
            2.3.1 item
            2.3.2 item
            2.3.3 item
        2.4 item
    3 item
    4 item

Note: IE8 supports these properties only if a !DOCTYPE is specified.
```

```
div {
  border: 3px solid red;
  padding: 5px;
  background-color:pink;
  font-size:20px;
}

img{   float: right;
  border: 3px solid blue;
p{ font-size:20px;
clear:right; }
.jtp{   overflow: auto; }
…… … … ….
<h1>Without Clearfix</h1>
<div>
  <img class="img1" src="jtp.png">
  Welcome to the javaTpoint.com. Here, you can
find the best tutorials that are easy to read and
help you to clear your concepts.
</div>
<p>Use the overflow:auto; CSS property</p>
<h1>With Clearfix</h1>
<div class="jtp">
  <img class="img2" src="jtp.png">
  Welcome to the javaTpoint.com. Here, you can
find the best tutorials that are easy to read and
help you to clear your concepts.
</div>
```

- A **clear float** (or **clearfix**) is a way for an element to fix or clear the child elements so that we do not require to add additional markup
- It resolves the error which occurs when more than one floated elements are stacked next to each other


Without Clearfix

Welcome to the javaTpoint.com. Here, you can find the best tutorials that are easy to read and help you to clear your concepts.

Use the overflow:auto; CSS property

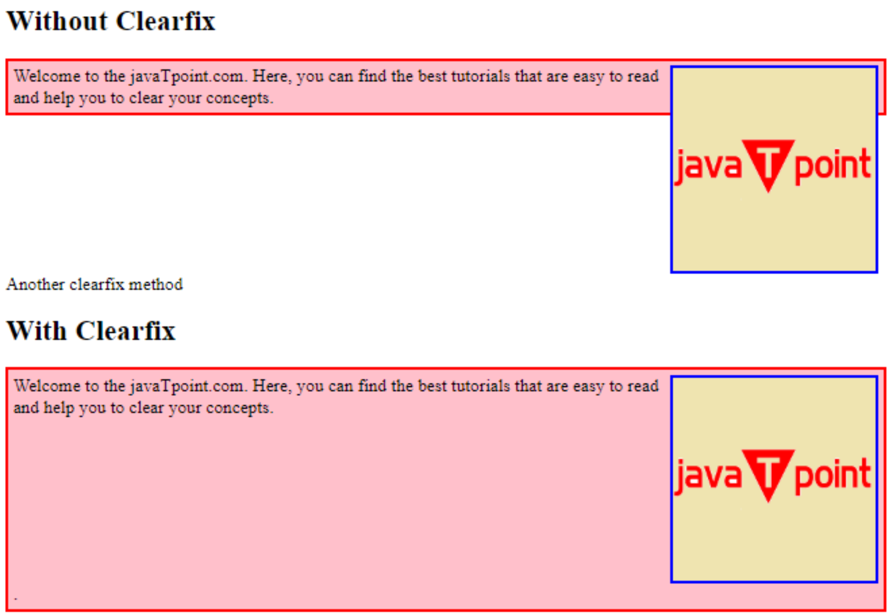
With Clearfix

Welcome to the javaTpoint.com. Here, you can find the best tutorials that are easy to read and help you to clear your concepts.

```
....
div {  border: 3px solid red;
  padding: 5px;
  background-color:pink;
  font-size:20px; }
img{  float: right;
  border: 3px solid blue; }
p{ font-size:20px;
clear:right; }
.jtp:after{ content:'.';
clear:both;
display: table; }
....
<h1>Without Clearfix</h1>
<div>
  <img src="jtp.png">
  Welcome to the javaTpoint.com. Here, you can find the best
tutorials that are easy to read and help you to clear your
concepts.
</div>
<p>Another clearfix method</p>
<h1>With Clearfix</h1>
<div class="jtp">
  <img src="jtp.png">
  Welcome to the javaTpoint.com. Here, you can find the best
tutorials that are easy to read and help you to clear your
concepts.
</div>
```

Another clearfix method

- In this example, we set the clear property to 'both', which stands that the floating elements will not allow on both the left and right sides
- We set the display property to 'table', which makes the element behave like <table> element of HTML. And we also have to leave the content empty



Without Clearfix

Welcome to the javaTpoint.com. Here, you can find the best tutorials that are easy to read and help you to clear your concepts.

Another clearfix method

With Clearfix

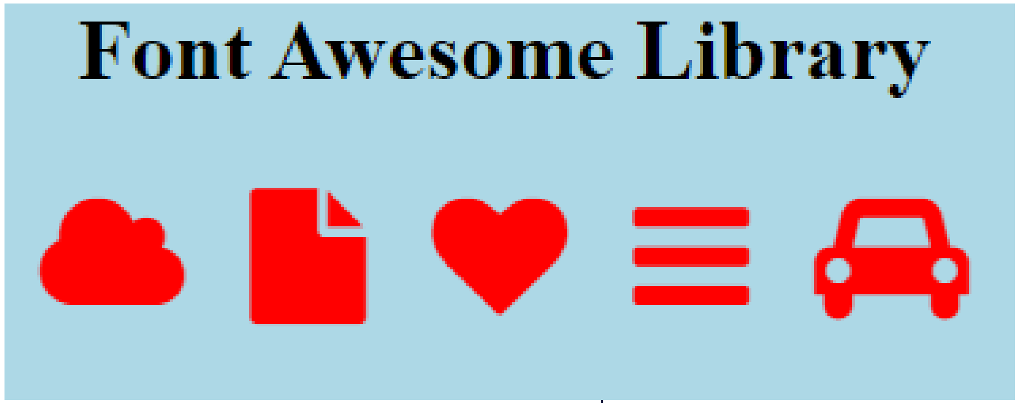
Welcome to the javaTpoint.com. Here, you can find the best tutorials that are easy to read and help you to clear your concepts.

CSS icons

- Icons can be defined as the images or symbols used in any computer interface refer to an element
- Using the icon library is the easiest way to add icons to our HTML page
  - Eg. Bootstrap icons, Font Awesome icons, and Google icons
- It is possible to format the library icons by using CSS
- There is no need to install or download the libraries mentioned

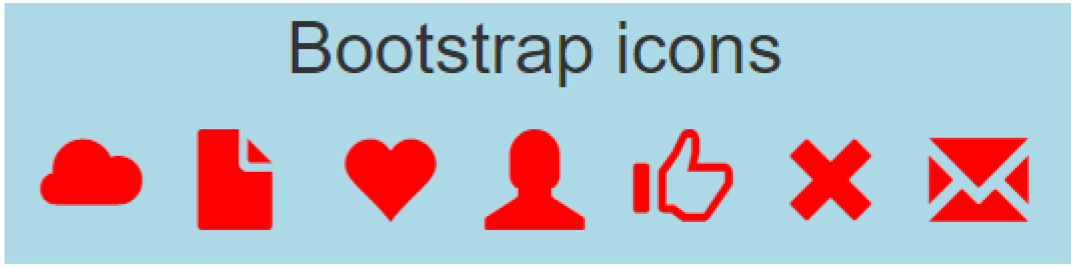
```
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
<style>
body{ text-align:center;
background-color:lightblue; }
.fa{ color:red;
font-size:50px;
margin:10px; }
……
<h1>Font Awesome Library</h1>
    <i class="fa fa-cloud"></i>
    <i class="fa fa-file"></i>
    <i class="fa fa-heart"></i>
    <i class="fa fa-bars"></i>
    <i class="fa fa-car"</i>
```

Example: using font Awesome icons


Font Awesome Library

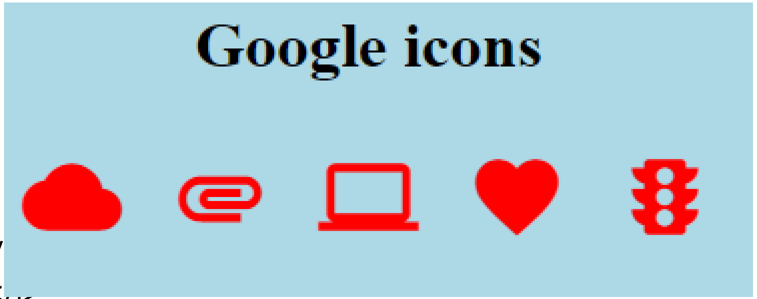
```
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
<style>
body{ text-align:center;
background-color:lightblue; }
.glyphicon {
color:red;
font-size:50px;
margin:10px; }
</style>
</head>
<h1>Bootstrap icons</h1>
    <i class="glyphicon glyphicon-cloud"></i>
    <i class="glyphicon glyphicon-file"></i>
    <i class="glyphicon glyphicon-heart"></i>
    <i class="glyphicon glyphicon-user"></i>
    <i class="glyphicon glyphicon-thumbs-up"></i>
    <i class="glyphicon glyphicon-remove"></i>
    <i class="glyphicon glyphicon-envelope"></i>
```

Example: using Bootstrap icons



```
………
<link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">
<style>
body{ text-align:center;
background-color:lightblue; }
.material-icons{ color:red;
font-size:50px;
margin:10px; }
………
<body>
<h1>Google icons</h1>
    <i class="material-icons">cloud</i>  <i class="material-icons">attachment</
    <i class="material-icons">computer</i>   <i class="material-icons">favorite</i>
    <i class="material-icons">traffic</i>
```

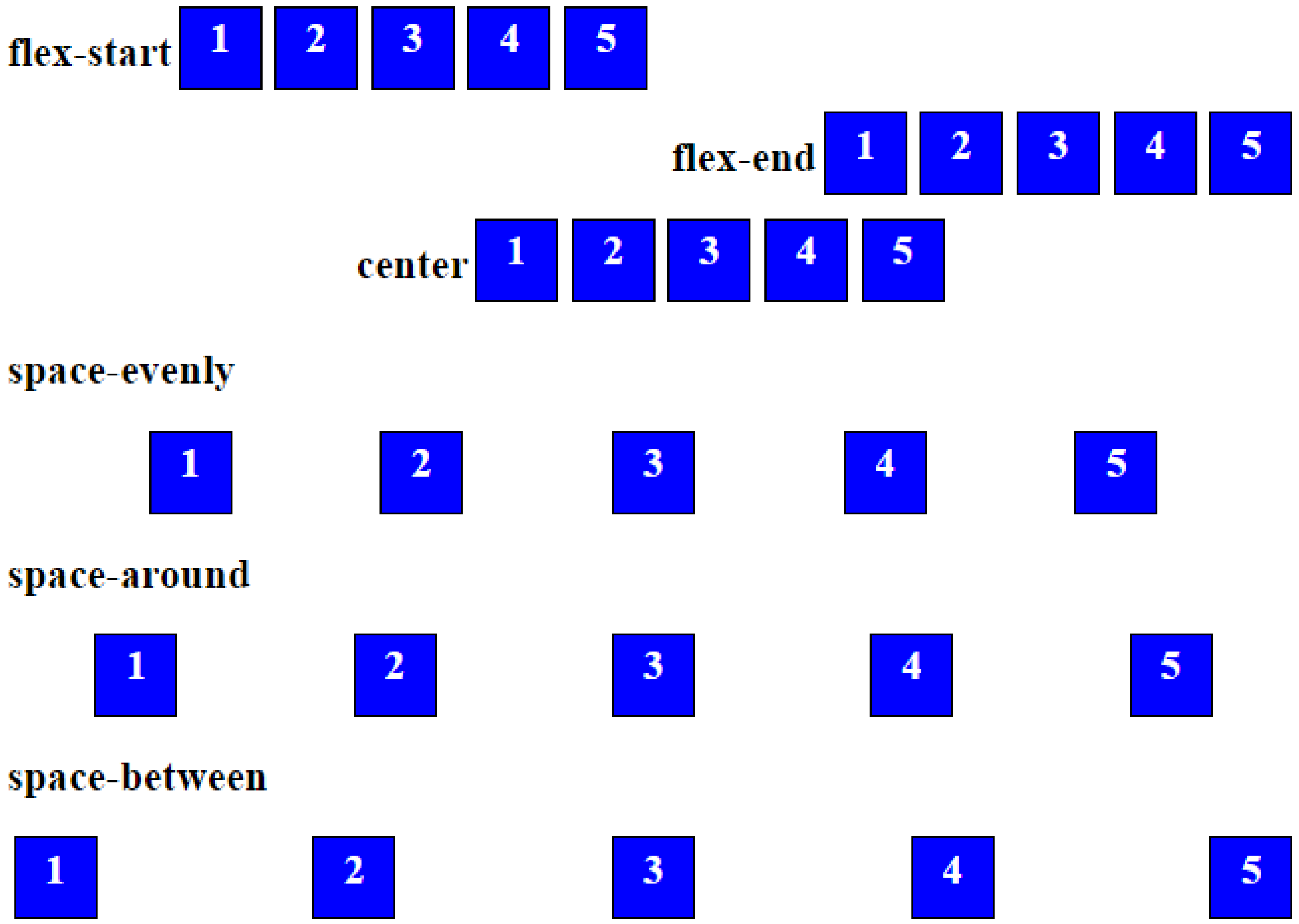
Example: Google icons



justify-content

- **Justify-content** property is used to align the items of the flexible box container when the items do not use all available space on the main-axis (horizontally)
- It defines how the browser distributes the space around and between the content items
- Syntax
  - justify-content: center | flex-start | flex-end | space-around | space-evenly | space-between | initial | inherit;
- The default value of this property is flex-start
- **center**: As its name implies, it set the position of items at the center of the container
- **flex-start**: It is the default value that positioned the items at the beginning of the container
- **flex-end**: It set the position of items at the end of the container.
- **space-around**: It positioned the items with equal spacing from each other. It evenly distributes the items in the line along with the same space around them
- **space-between**: Items are evenly spaced in which the first item is at the beginning, and the last item is at the end
- **space-evenly**: It also positioned the items with equal space, but the spacing from the corners differs

justify-content example



CSS Lists

- The CSS properties to style the lists are given as follows:
  - **list-style-type**: This property is responsible for controlling the appearance and shape of the marker
  - **list-style-image**: It sets an image for the marker instead of the number or a bullet point
  - **list-style-position**: It specifies the position of the marker.
  - list-style: It is the shorthand property of the above properties
  - **marker-offset**: It is used to specify the distance between the text and the marker. It is unsupported in IE6 or Netscape 7
- list-style-type:

```
…….
<style>
        .num { list-style-type:decimal; }
        .alpha{ list-style-type:lower-alpha; }
        .roman{ list-style-type:lower-roman; }
        .circle{ list-style-type:circle; }
        .square{ list-style-type:square;  }
        .disc{ list-style-type:disc;  }
………
```

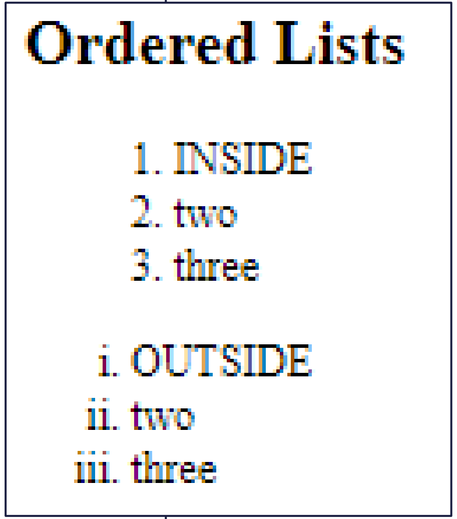
```
<h2>  Ordered Lists </h2>
    <ol class="num">
      <li>one</li>
      <li>two</li>
      <li>three</li>
    </ol>
    <ol class="alpha">
      <li>one</li>
      <li>two</li>
      <li>three</li>
    </ol>
        <ol class="roman">
      <li>one</li>
      <li>two</li>
      <li>three</li>
    </ol>
```

```
<h2>  Unordered lists </h2>
    <ul class="disc">
      <li>one</li>
      <li>two</li>
      <li>three</li>
    </ul>
    <ul class="circle">
      <li>one</li>
      <li>two</li>
      <li>three</li>
    </ul>
   <ul class="square">
      <li>one</li>
      <li>two</li>
      <li>three</li>
    </ul>
```

CSS Lists cont’d

- **list-style-position** It represents whether the appearing of the marker is inside or outside of the box containing the bullet points
  - **inside**: It means that the bullet points will be in the list item
    - In this, if the text goes on the second line, then the text will be wrap under the marker
  - **outside**: It represents that the bullet points will be outside the list item. It is the default value

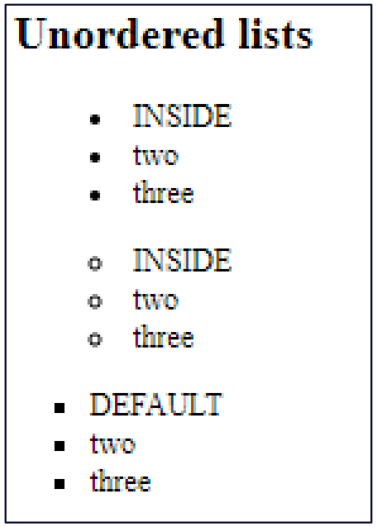
```
<style>
        .num{  list-style-type:decimal;
               list-style-position:inside;   }
        .roman{  list-style-type:lower-roman;
         list-style-position:outside;  }
        .circle{   list-style-type:circle;
              list-style-position:inside;  }
        .square{ list-style-type:square;   }
        .disc{  list-style-type:disc;
              list-style-position:inside;  }
</style>
```



Ordered Lists

1. INSIDE
2. two
3. three

i. OUTSIDE
ii. two
iii. three



Unordered lists

- INSIDE
- two
- three

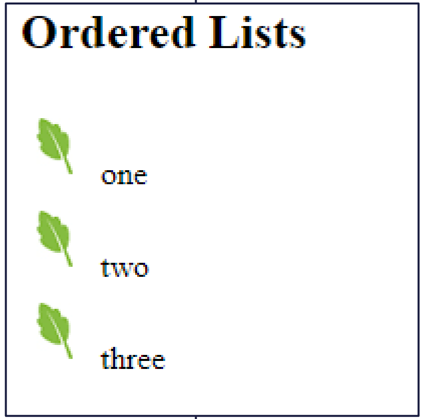
- INSIDE
- two
- three

- DEFAULT
- two
- three

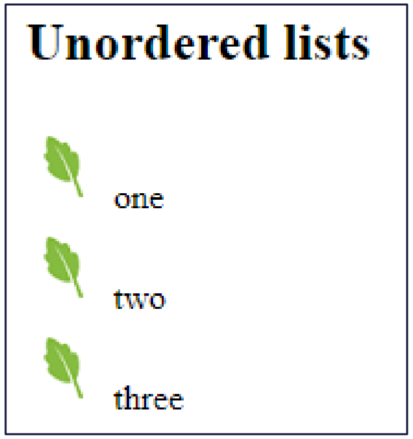
list-style-image

- It specifies an image as the marker
- Its syntax is similar to the background-image property. If it does not find the corresponding image, the default bullets will be used

```
<style>
        .order{
          list-style-image: url(img.png);
        }
        .unorder{
          list-style-image: url(img.png);
        }

    </style>
```


Ordered Lists

- one
- two
- three

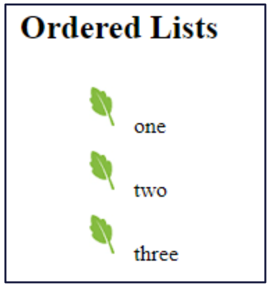

Unordered lists

- one
- two
- three

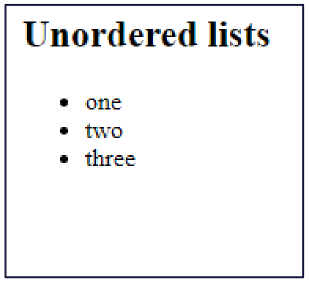
list-style

- **list-style** is the shorthand property that is used to set all list properties in one expression
  - The order of the values of this property is type, position, and image. But if any property value is missing, then the default value will be inserted

```
<style>
.order{ list-style: lower-alpha inside url(img.png);  }
.unorder{ list-style: disc outside; }
```

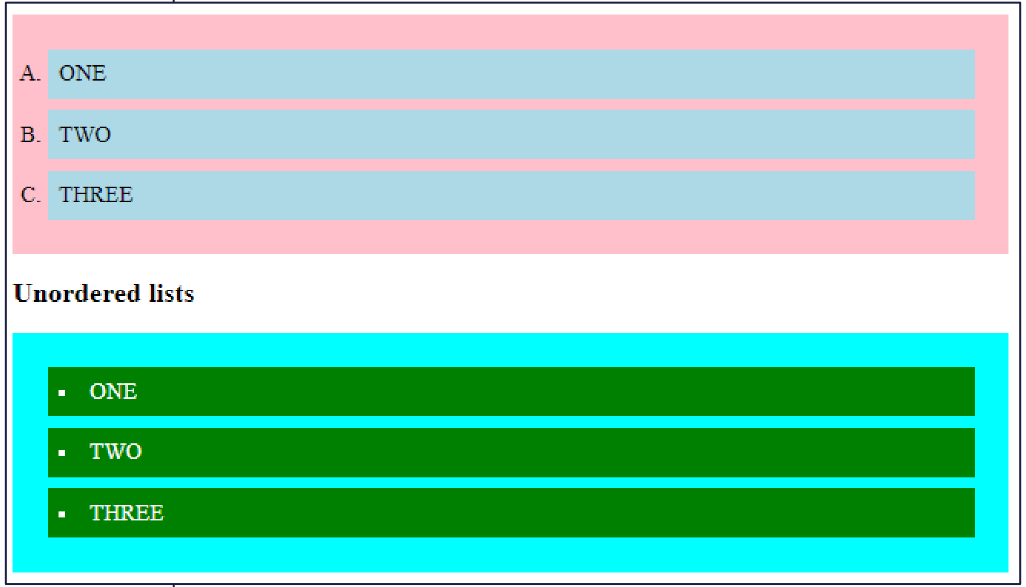

Ordered Lists

one
two
three


Unordered lists

- one
- two
- three

```
        .order{ list-style: upper-alpha;
              background: pink;
              padding:20px;     }
        .order li{    background: lightblue;
        padding:10px;
        font-size:20px;
        margin:10px;      }
        .unorder{ list-style: square inside;
              background: cyan;
              padding:20px;     }
        .unorder li{  background: green;
        color:white;
        padding:10px;
        font-size:20px;
        margin:10px;      }
```


A. ONE
B. TWO
C. THREE

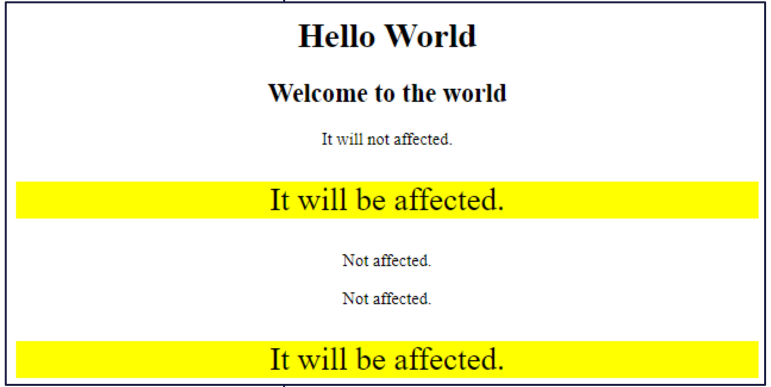
Unordered lists

▪ ONE
▪ TWO
▪ THREE

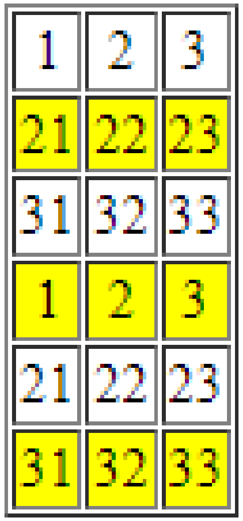
CSS :nth-child(n) selector

- **:nth-child(n)** selector is used for matching the nth-child elements based on their position regardless of the type of its parent
  - The n can either be a keyword, formula, or a number
  - It is used to match the elements based on their position within a group of siblings
- Syntax
  - :nth-child(n) { //CSS Property }
- **Functional notation (An+B):** Functional notation represents those elements whose siblings position match **the An+B** pattern
  - where **A** is the integer step size, **n** is any positive integer starting from 0, and **B** is the integer offset

```
p:nth-child(3n+4) {  background: yellow;
            font-size:30px;  }
 tr:nth-child(2n) { background: green; }
.....
 <body style = "text-align:center">
      <h1> Hello World </h1>
      <h2>  Welcome to the javaTpoint  </h2>
      <p>It will not affected.</p>
      <p>It will be affected.</p>
      <p>Not affected.</p>
      <p>Not affected.</p>
      <p>It will be affected.</p>
       <table border=1>
         <tr> <td>1</td> <td>2</td> <td>3</td></tr>
         <tr> .....
```



```
Hello World
Welcome to the world
It will not affected.
It will be affected.
Not affected.
Not affected.
It will be affected.
```

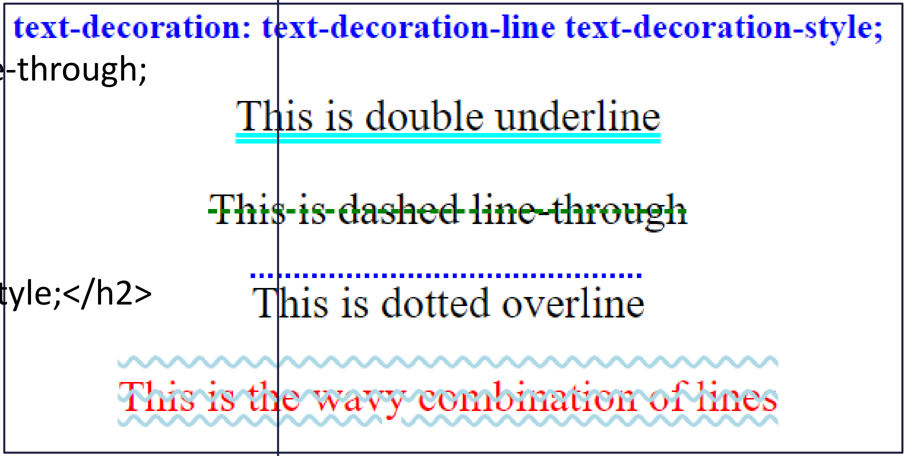


1	2	3
21	22	23
31	32	33
1	2	3
21	22	23
31	32	33

text-decoration

- **text-decoration** property that decorates the content of the text
  - This is shorthand for text-decoration-line, text-decoration-color, and text-decoration-style
- Syntax
  - text-decoration: text-decoration-line text-decoration-color text-decoration-style|initial|inherit;

```
…….
#p1 { text-decoration: underline double cyan; }
#p2 { text-decoration: line-through dashed green; }
#p3 { text-decoration: dotted overline blue; }
#p4 { text-decoration: lightblue wavy overline underline line-through;
        color:red; }
</style>
….
<body>
<h2>text-decoration: text-decoration-line text-decoration-style;</h2>
   <div>
      <p id="p1">This is double underline</p>
      <p id="p2">This is dashed line-through</p>
      <p id="p3">This is dotted overline</p>
        <p id="p4">This is the wavy combination of lines</p>
…
```



text-decoration: text-decoration-line text-decoration-style;

This is double underline

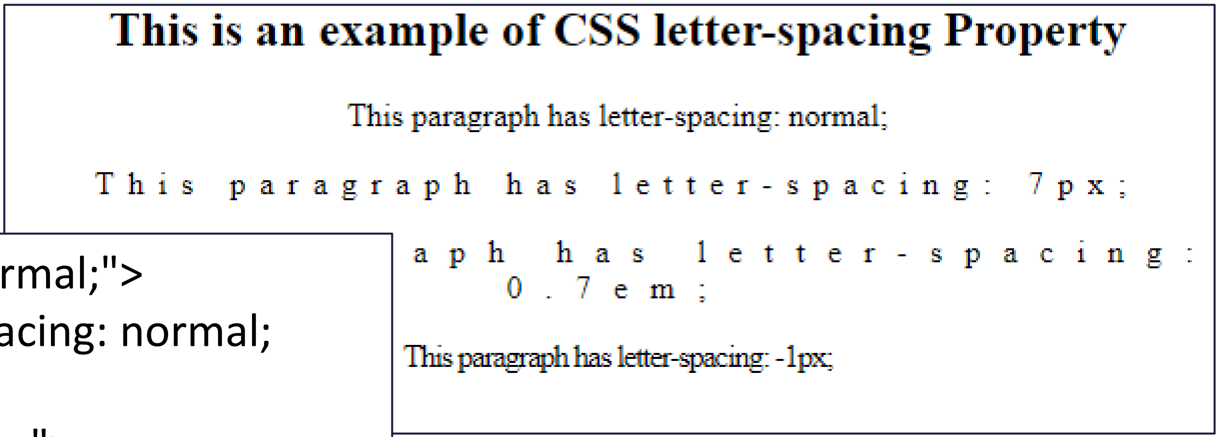
~~This is dashed line-through~~

This is dotted overline

~~This is the wavy combination of lines~~

CSS letter-spacing

- The letter-spacing property used to control the space between every letter inside an element or the block of text
- Syntax
  - letter-spacing: normal | length | initial | inherit;



This is an example of CSS letter-spacing Property

This paragraph has letter-spacing: normal;

This paragraph has letter-spacing: 7px;

aph has letter-spacing: 0.7em;

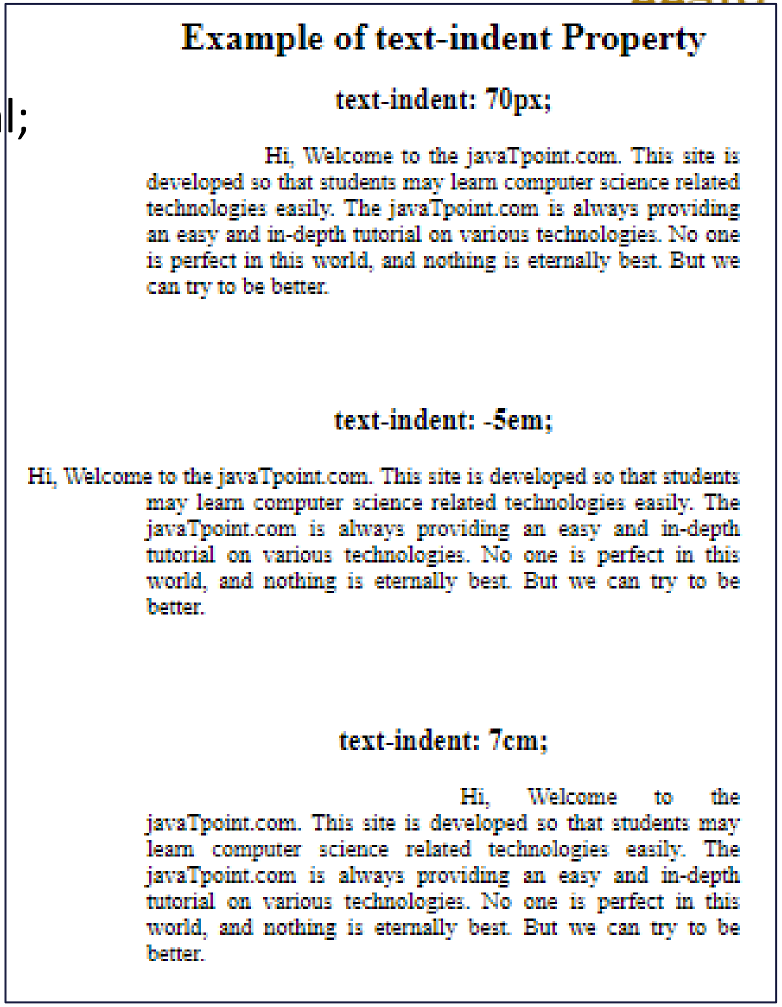
This paragraph has letter-spacing: -1px;

```
<p style= "letter-spacing: normal;">
This paragraph has letter-spacing: normal;
</p>
<p style= "letter-spacing: 7px;">
This paragraph has letter-spacing: 7px; </p>
<p style= "letter-spacing: 0.7em;">
This paragraph has letter-spacing: 0.7em; </p>
<p style= "letter-spacing: -1px;">
This paragraph has letter-spacing: -1px; </p>
```

CSS text-indent

- This CSS property sets the indentation of the first line in a block of text
- Syntax
  - **text-indent**: length | inherit | initial;

```
      div{   font-size: 20px;
             width: 500px;
             height:200px;
             text-align: justify; }
       .jtppx {text-indent: 100px; }
      .jtpem { text-indent: -5em;  }
      .jtpcm { text-indent: 7cm;  }
……
     <h1>Example of text-indent Property</h1>
     <h2>text-indent: 70px;</h2>
<div class = "jtppx">  Hi, Wel… so that ….   </div>
      <h2>text-indent: -5em;</h2>
<div class = "jtpem"> Hi, We ….computer ..  </div>
     <h2>text-indent: 7cm;</h2>
<div class = "jtpcm"> Hi, Welcome …learn   </div>
```



Example of text-indent Property

text-indent: 70px;

Hi, Welcome to the javaTpoint.com. This site is developed so that students may learn computer science related technologies easily. The javaTpoint.com is always providing an easy and in-depth tutorial on various technologies. No one is perfect in this world, and nothing is eternally best. But we can try to be better.

text-indent: -5em;

Hi, Welcome to the javaTpoint.com. This site is developed so that students may learn computer science related technologies easily. The javaTpoint.com is always providing an easy and in-depth tutorial on various technologies. No one is perfect in this world, and nothing is eternally best. But we can try to be better.

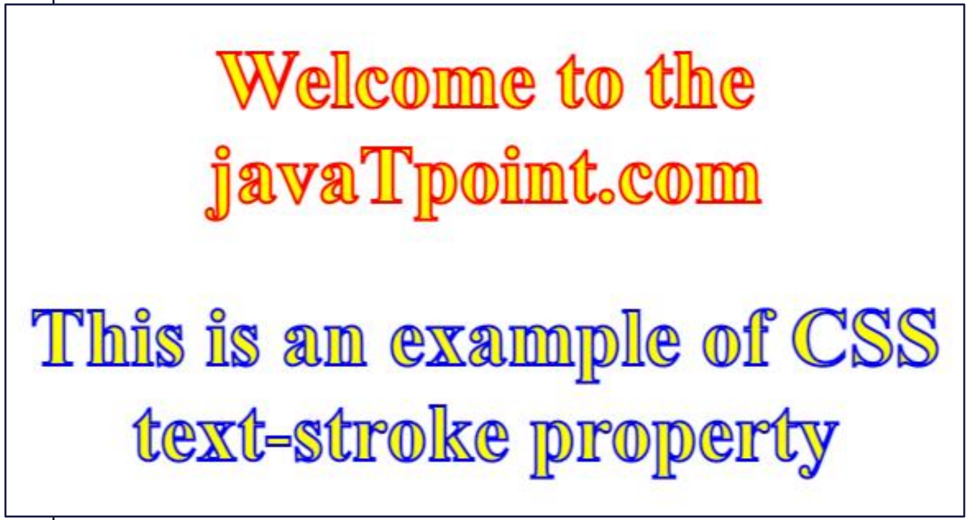
text-indent: 7cm;

Hi, Welcome to the javaTpoint.com. This site is developed so that students may learn computer science related technologies easily. The javaTpoint.com is always providing an easy and in-depth tutorial on various technologies. No one is perfect in this world, and nothing is eternally best. But we can try to be better.

CSS text-stroke

- This CSS property adds a stroke to the text and also provides decoration options for them
  - It is the shorthand of the **text-stroke-width** and **text-stroke-color**
- The text-stroke can only be used with the -webkit- prefix

```
      .jtp {
        color: yellow;
        font-size: 50px;
        -webkit-text-stroke-width: 2px;
        -webkit-text-stroke-color: red;
      }
.....
<h1 class= "jtp">Welcome to the
javaTpoint.com</h1>
<h2 class= "jtp" style= "-webkit-text-
stroke-color: blue;">This is an example
of CSS text-stroke property</h2>
</body>
```

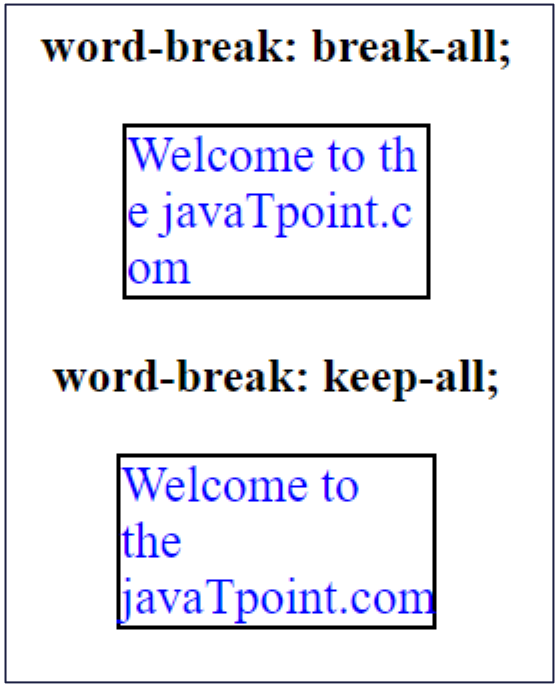

Welcome to the javaTpoint.com
This is an example of CSS text-stroke property

CSS Text Effects

- **word-break:** specifies how words should break at the end of the line. It defines the line break rules
- Syntax
  - word-break: normal |keep-all | break-all | inherit ;
  - The default value of this property is normal
  - **keep-all**: It breaks the word in the default style
  - **break-all**: It inserts the word break between the characters in order to prevent the word overflow

```
       .jtp{
           width: 150px;
           border: 2px solid black;
           word-break: break-all;
           text-align: left;
           font-size: 25px;
        color: break;
         }
.....
      <h2>word-break: break-all;</h2>
 <p class="jtp"> Welcome to the javaTpoint.com</p>
      <h2>word-break: keep-all;</h2>
<p class="jtp1"> Welcome to the javaTpoint.com </p>
```

```
.jtp1{
          width: 156px;
          border: 2px solid black;
          word-break: keep-all;
          text-align: left;
          font-size: 25px;
      color: blue;
       }
```


word-break: break-all;

Welcome to th
e javaTpoint.c
om

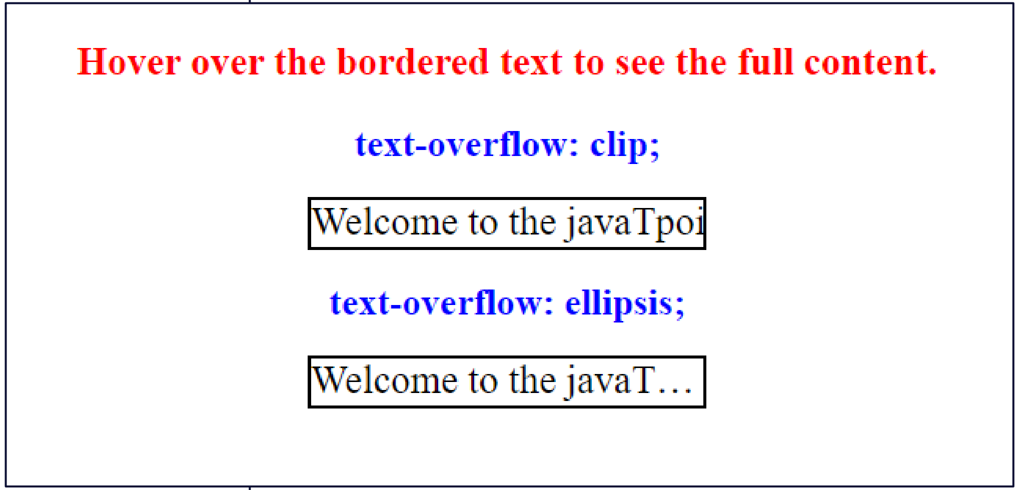
word-break: keep-all;

Welcome to
the
javaTpoint.com

text-overflow

- text-overflow specifies the representation of overflowed text, which is not visible to the user
- This property does not work on its own. We have to use white-space: nowrap; and overflow: hidden; with this property
- Syntax
  - text-overflow: clip | ellipsis;
  - **clip**: It is the default value that clips the overflowed text
  - **ellipsis**: This value displays an ellipsis (…) or three dots to show the clipped text

```
.jtp{    white-space: nowrap;
           height: 30px;
           width: 250px;
           overflow: hidden;
           border: 2px solid black;
           font-size: 25px;
           text-overflow: clip;  }
.jtp1 { white-space: nowrap;
          height: 30px;
           width: 250px;
           overflow: hidden;
           border: 2px solid black;
           font-size: 25px;
           text-overflow: ellipsis;   }
   …..
<p> Hover over the bordered text to see the full content. </p>
<h2> text-overflow: clip;  </h2>
 <div class="jtp"> Welcome to the javaTpoint.com </div>
     <h2>  text-overflow: ellipsis;  </h2>
<div class="jtp1">  Welcome to the javaTpoint.com  </div>….
```


Hover over the bordered text to see the full content.

text-overflow: clip;

Welcome to the javaTpoi

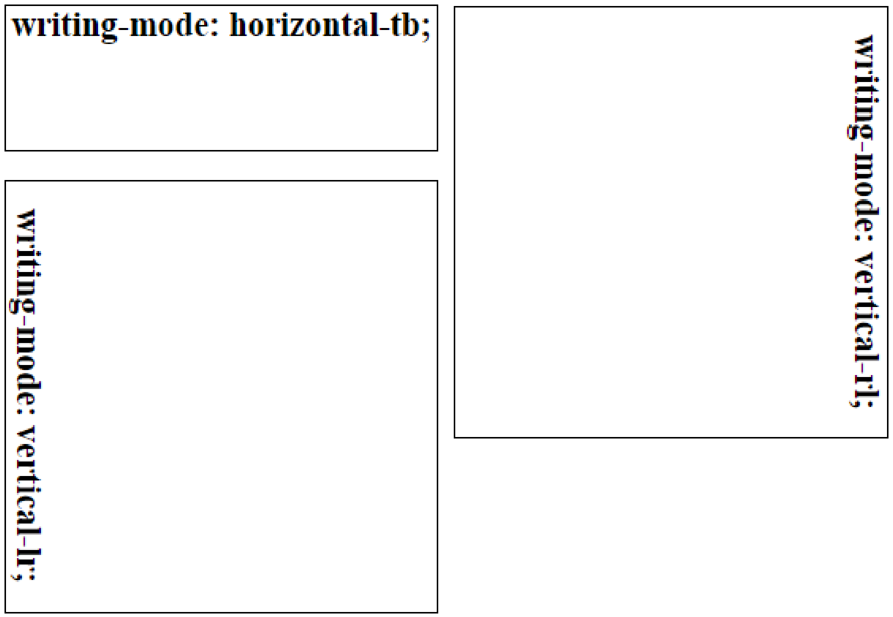
text-overflow: ellipsis;

Welcome to the javaT…

CSS writing-mode

- writing-mode: specifies whether the text will be written in the horizontal or vertical direction
- Syntax
  - writing-mode: horizontal-tb | vertical-lr | vertical-rl | inherit ;
  - **horizontal-tb**: It is the default value of this property in which the text is in the horizontal direction and read from left to right and top to bottom
  - **vertical-rl**: It displays the text in the vertical direction, and the text is read from right to left and top to bottom.
  - **vertical-lr**: It is similar to vertical-rl, but the text is read from left to right.

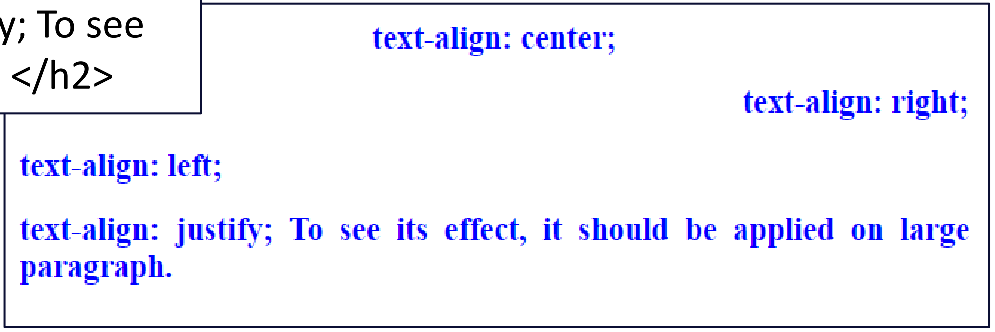
```
….
   h2 {   border: 2px solid black;
     width: 300px;
     height: 100px;    }
   #tb {  writing-mode: horizontal-tb;    }
   #lr { writing-mode: vertical-lr;  }
   #rl { writing-mode: vertical-rl; }
  ….
  <h1> Example of CSS writing-mode property </h1>
  <h2 id="tb"> writing-mode: horizontal-tb; </h2>
  <h2 id="lr" style= "height: 300px;"> writing-mode:
vertical-lr; </h2><br>
  <h2 id="rl" style= "height: 300px;"> writing-mode:
vertical-rl; </h2>…..
```


writing-mode: horizontal-tb;
writing-mode: vertical-lr;
writing-mode: vertical-rl;

CSS text-align

- **text-align** property is used to set the horizontal alignment of a table-cell box or the block element
- Syntax
  - text-align: justify | center | right | left | initial | inherit;
  - justify: It is generally used in newspapers and magazines. It stretches the element's content in order to display the equal width of every line.
  - center: It centers the inline text.
  - right: It is used to align the text to the right
  - left: It is used to align the text to the left

```
....
  <h1>Example of text-align proeprty</h1>
<h2 style = "text-align: center;"> text-align: center; </h2>

<h2 style = "text-align: right;"> text-align: right;</h2>
<h2 style = "text-align: left;">text-align: left; </h2>
 <h2 style = "text-align: justify;"> text-align: justify; To see
its effect, it should be applied on large paragraph </h2>
```



text-align: center;

text-align: right;

text-align: left;

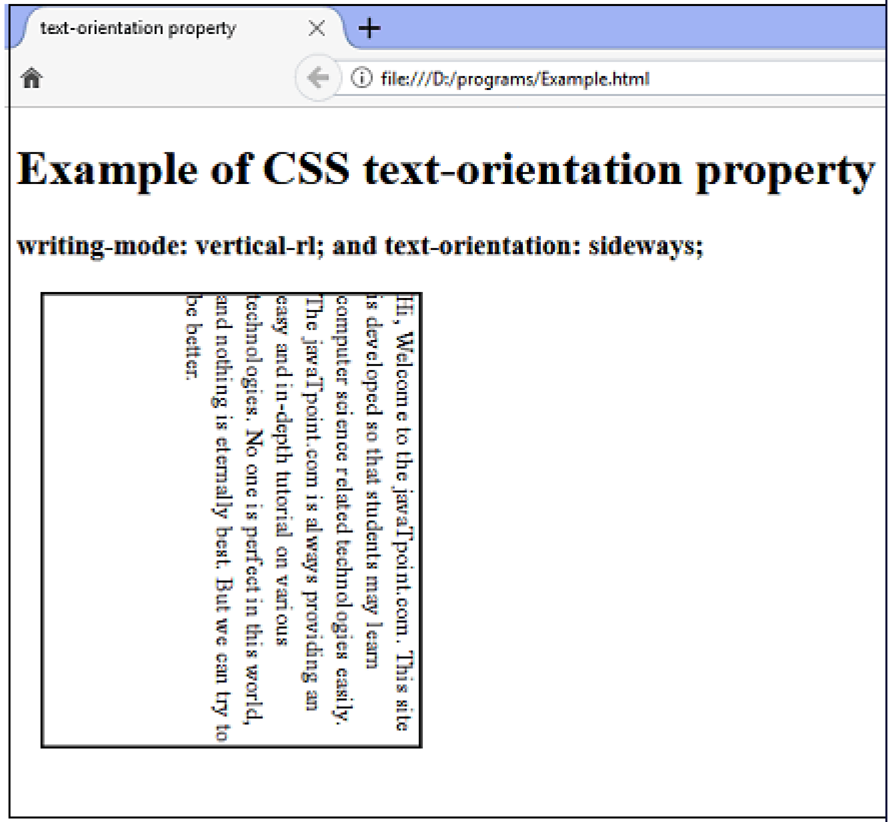
text-align: justify; To see its effect, it should be applied on large paragraph.

CSS text-orientation

- **text-orientation** property specifies the orientation of characters in the line of content
  - It only applies to the vertical mode of content. This property does not affect elements with horizontal writing mode
- Syntax
  - text-orientation: mixed | upright | sideways | sideways-right | use-glyph-orientation | initial | inherit;

Values	Description
mixed	It is the default value that rotates the characters 90o degree clockwise. It set the characters of vertical script naturally.
upright	This value sets the characters of horizontal scripts naturally (upright), as well as the glyphs for the vertical scripts. It makes all characters to be considered as left-to-right.
sideways	It rotates the line to 90o clockwise. This value causes the characters to be laid out as horizontally. This value does not work in Google Chrome and other major browsers except Firefox, i.e., it only works in Firefox.
sideways-right	It is kept for compatibility purposes. It is an alias to the value sideways.
use-glyph-orientation	This value is not used anymore.
initial	It sets the property to its default value.
inherit	It inherits the property from its parent element.

CSS text-orientation example



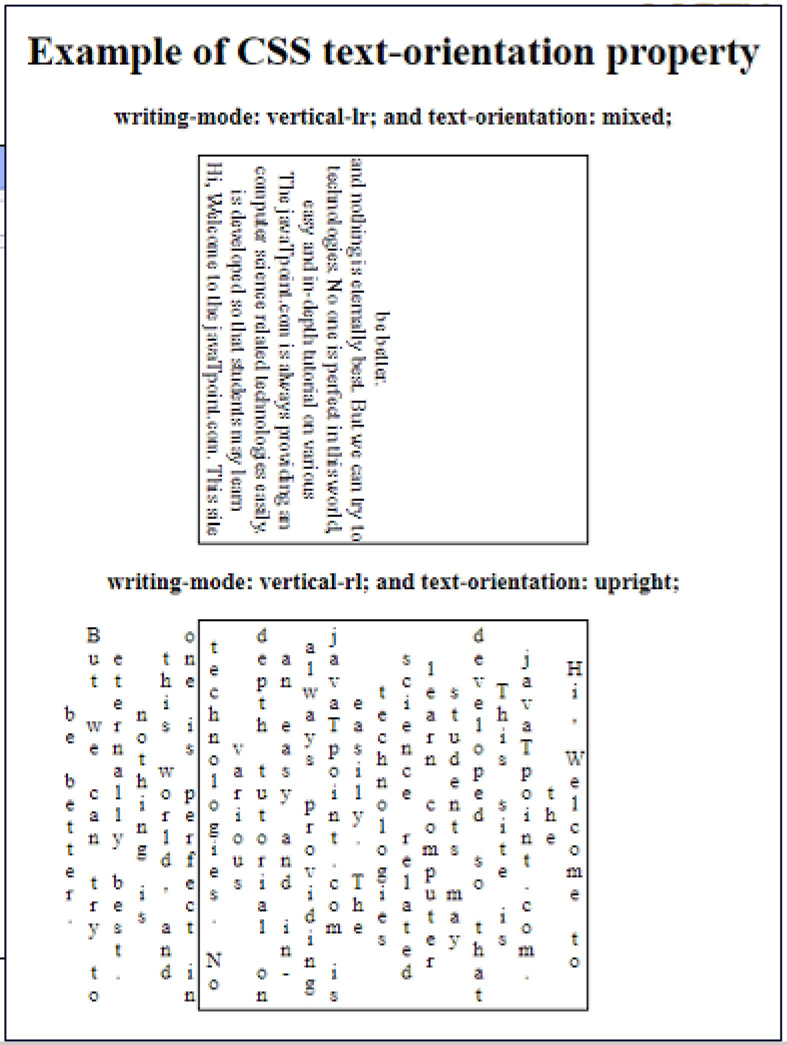
text-orientation property

file:///D:/programs/Example.html

Example of CSS text-orientation property

writing-mode: vertical-rl; and text-orientation: sideways;

Hi, Welcome to the javaTpoint.com. This site is developed so that students may learn computer science related technologies easily. The javaTpoint.com is always providing an easy and in-depth tutorial on various technologies. No one is perfect in this world, and nothing is eternally best. But we can try to be better.



Example of CSS text-orientation property

writing-mode: vertical-lr; and text-orientation: mixed;

Hi, Welcome to the javaTpoint.com. This site is developed so that students may learn computer science related technologies easily. The javaTpoint.com is always providing an easy and in-depth tutorial on various technologies. No one is perfect in this world, and nothing is eternally best. But we can try to be better.

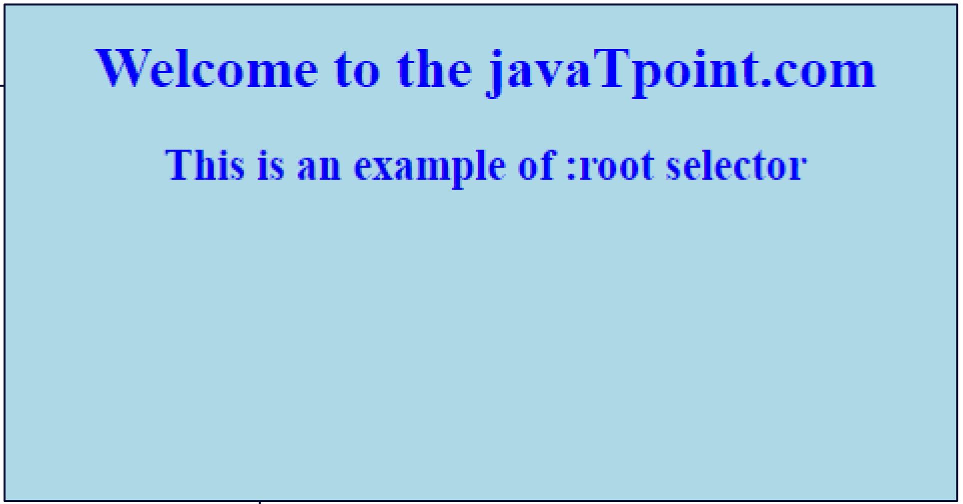
writing-mode: vertical-rl; and text-orientation: upright;

Hi, Welcome to the javaTpoint.com. This site is developed so that students may learn computer science related technologies easily. The javaTpoint.com is always providing an easy and in-depth tutorial on various technologies. No one is perfect in this world, and nothing is eternally best. But we can try to be better.

CSS :root Selector

- This pseudo-class in CSS matches the root element of the document
  - It selects the highest-level parent in the document tree or DOM
- In HTML, the <html> element is always the root element. Although the :root is equivalent to html selector because both of them target the same element, but the :root selector has a higher specificity
- Syntax
  - :root { // CSS property }

```
…
      :root {
          background: lightblue;
          color: blue;
          text-align: center;
        }
    </style>
   </head>
   <body>
      <h1>Welcome to the javaTpoint.com</h1>
      <h2>This is an example of :root selector</h2>
   </body>
```



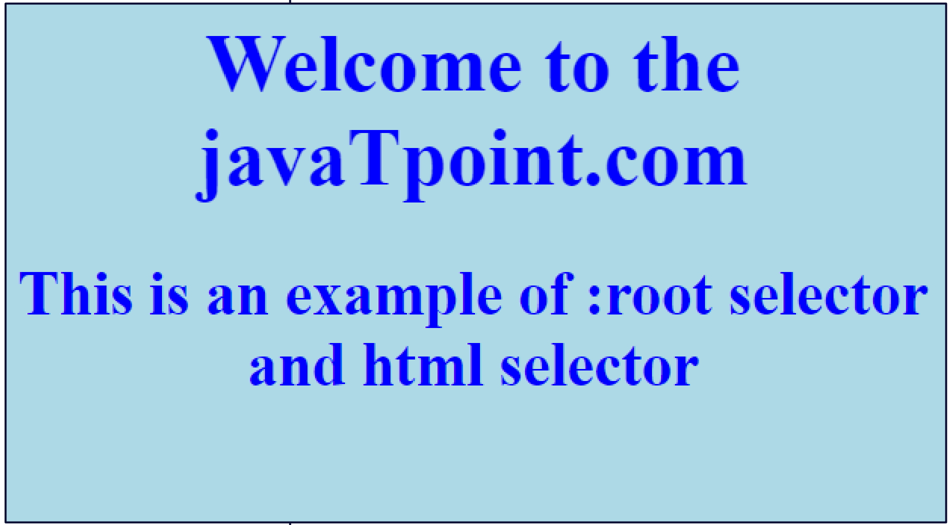
Welcome to the javaTpoint.com

This is an example of :root selector

CSS :root Selector example

In this example, we can see that the **background-color** and **color** properties are both mentioned in **html** as well as in **:root** selector, but in the output, the properties mentioned in **:root** selector will work. This is because of the higher specificity of the **:root** selector

```
.....
      <style>
        :root {  background-color: lightblue;
           color: blue;
           text-align: center; }
        html { background-color: red;
        color: white;
        font-size: 30px; }
      </style>
   </head>
   <body>
      <h1>Welcome to the javaTpoint.com</h1>
      <h2>This is an example of :root selector and html
selector</h2>
....
```



Welcome to the javaTpoint.com

This is an example of :root selector and html selector

CSS calc()

- It is an inbuilt CSS function which allows us to perform calculations
- It can be used to calculate length, percentage, time, number, integer frequency, or angle
- It uses the four simple arithmetic operators add (+), multiply (*), subtract (-), and divide (/)
- Allows us to mix any units, such as percentage and pixel
- Syntax: calc(Expression);
- Important points
  - The arithmetic operators add (+) and subtract (-) should always be surrounded by whitespace. Otherwise, the expression will be treated as an invalid expression
    - Example, calc(60%-4px) will be invalid because it is parsed as a percentage, followed by a negative length. On the other hand, the expression calc(60% - 4px) will be parsed as a subtraction operator and a length
  - Although the operators * and / does not requires whitespace, but it is recommended to add it for consistency
  - Nesting of calc() function can be possible

CSS calc() example

```
<style>
        .jtp {
           width: calc(150% - 75%);
           height: calc(350px - 75px);
           background-color: lightblue;
           padding-top: 50px;
        }
        .jtp1 {
           font-size: 30px;
           font-weight: bold;
           color: blue;
        }
....
</style>
```

Using mixed unit

```
<style>
        .jtp {
           width: calc(40% + 10em);
           height: calc(350px + 75px);
           background-color: lightblue;
           padding-top: calc(10% - 10px);
          padding-left: calc(10% + 10px);
          font-size: 30px;
        }
…
</style>
```

Nested calc()

```
<style>
        .jtp {
           width: calc( calc(40em / 3) * 2);
           height: calc(350px + 75px);
           background-color: lightblue;
        }
</style>
```

CSS Zoom

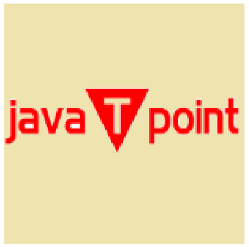
- This CSS property scales the content. It manages the magnification level of the content
  - Instead of using this property, we can also use the transform: scale(); property
- Syntax
  - zoom: normal | number | percentage | reset ;

```
….
.magnify1{  zoom: 0.75;  }
  .magnify{   zoom: 1.5;   }
 </style>
……
<body>
…
      <img class="magnify1" src= "jtp.png">
      <h2>Magnified image zoom: 1.5;</h2>
       <img class="magnify" src= "jtp.png">
```

original image

javaTpoint

Magnified image zoom: 0.75;

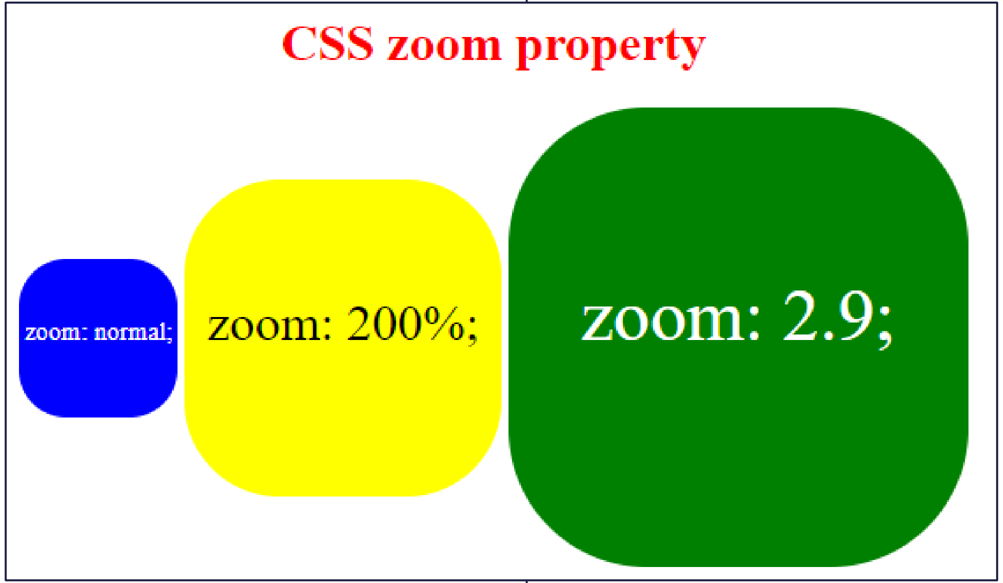


Magnified image zoom: 1.5;

javaTpoint

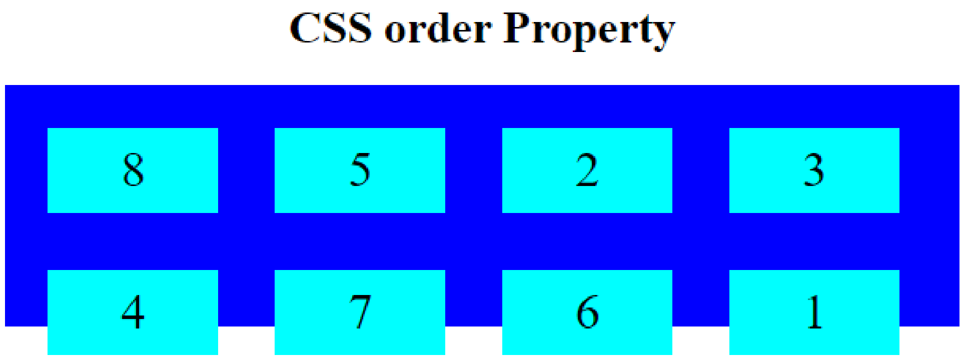
CSS Zoom cont'd

```
…
.magnify{  width: 100px;
  height: 100px;
  border-radius: 30px;
  display: inline-block;
  color: white; }
#m1 {  background-color: blue;
  zoom: normal; }
#m2 {  background-color: yellow;
  zoom: 200%;
  color: black; }
#m3 {   background-color: green;
  zoom: 2.9; }
p{ padding-top:20px; }
</style>
…
<body>
<div id="m1" class="magnify"><p>zoom: normal;<p></div>
<div id="m2" class="magnify"><p>zoom: 200%;<p></div>
<div id="m3" class="magnify"><p>zoom: 2.9;<p></div>
```



CSS order

- This CSS property specifies the order of the flex item in the grid container or flex
- It is basically used for ordering the flex items
- It is to note that, if the element isn't flexible, then this property will not work
- Syntax
  - order: integer | initial | inherit;

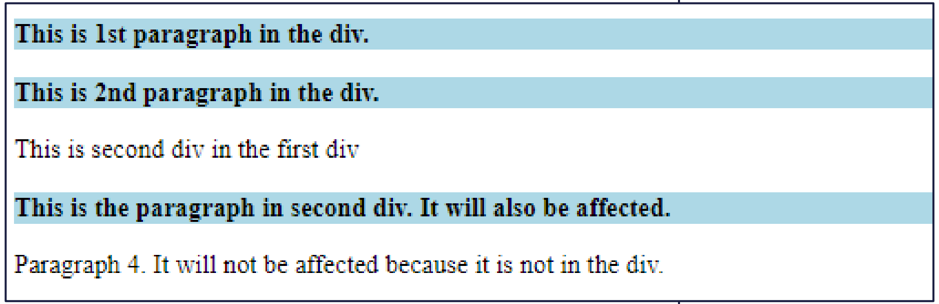


```
…
 .container {
         display: flex;
         background-color: blue;
         height: 150px;
         width: auto;
         flex-wrap: wrap;  }
…..
    </style>
  </head>
  <body>
  <h1> CSS order Property </h1>
     <div class = "container">
       <div style = "order: 3"> 1 </div>
       <div style = "order: 0"> 2 </div>
       <div style = "order: 0"> 3 </div>
       <div style = "order: 1"> 4 </div>
       <div style = "order: -1"> 5 </div>
       <div style = "order: 2"> 6 </div>
       <div style = "order: 1"> 7 </div>
       <div style = "order: -2"> 8 </div>
    </div>
  </body>
</html>
```

CSS Descendant Selector

- The CSS descendant selector is used to match the descendant elements of a particular element
- Syntax
  - selector1 selector2 { /* property declarations */ }

```
<style>
div p {  background-color: lightblue;
  font-weight: bold; }
</style>
..
<div>
  <p> This is 1st paragraph in the div. </p>
  <p> This is 2nd paragraph in the div. </p>
  <div>
This is second div in the first div
  <p> This is the paragraph in second div. It will also be affected. </p>
</div>
</div>
<p> Paragraph 4. It will not be affected because it is not in the div. </p>
```



This is 1st paragraph in the div.

This is 2nd paragraph in the div.

This is second div in the first div

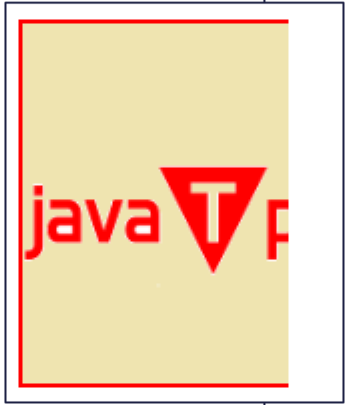
This is the paragraph in second div. It will also be affected.

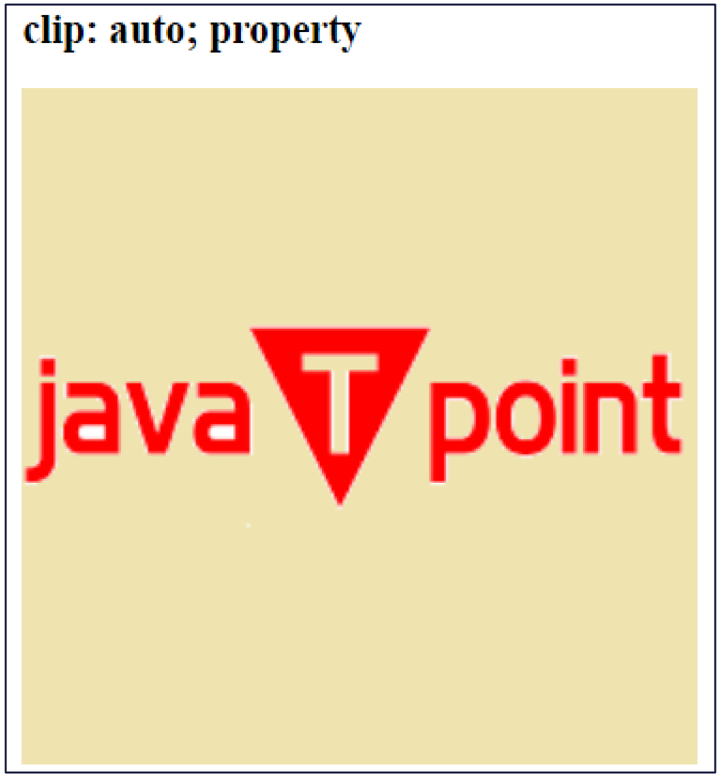
Paragraph 4. It will not be affected because it is not in the div.

CSS Clip

- This CSS property specifies the visible area of an element
  - It applies to absolutely positioned elements (position: absolute;)
- Syntax
  - clip: auto | shape | initial | inherit;

```
….
    <style type = "text/css">
      div {
        background: url(jtp.png);
        clip: rect(0px, 150px, 250px, 0px);
        border:3px solid red;
        height:200px;
        width: 250px;
        position: absolute;
      }
    </style>
….
  <body>
    <div></div>
…
```

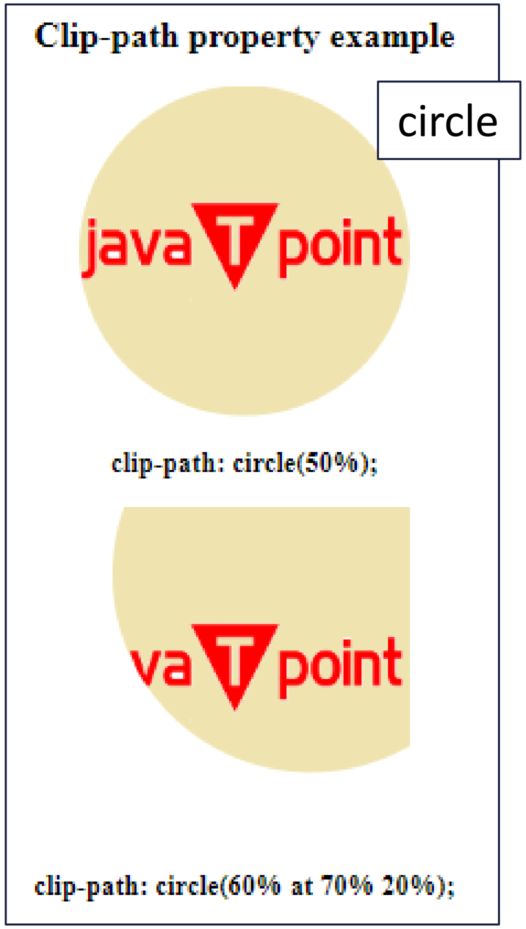



clip: auto; property

CSS clip-path

- This CSS property is used to create a clipping region and specifies the element's area that should be visible
  - The area inside the region will be visible, while the outside area is hidden
- Anything outside the clipping will be clipped by browsers, including borders, text-shadows, and many more.
- Syntax
  - clip-path: <clip-source> | [<basic-shape> || <geometry-box> || none
- **clip-source:** It is a url that reference to an SVG <clippath> element
- **basic-shape:** It clips the element to a basic shape. It has four basic shapes: **circle**, **ellipse**, **polygon** and **inset**.
- **geometry-box:** The <geometry-box> defines the reference box for the basic shape
  - If it is defined with the combination of the <basic-shape>, then it will act as the reference box for the clipping done by the <basic-shape>
- **geometry-box** can have the below values:
  - margin-box, border-box, padding-box, content-box, fill-box, stroke-box, view-box

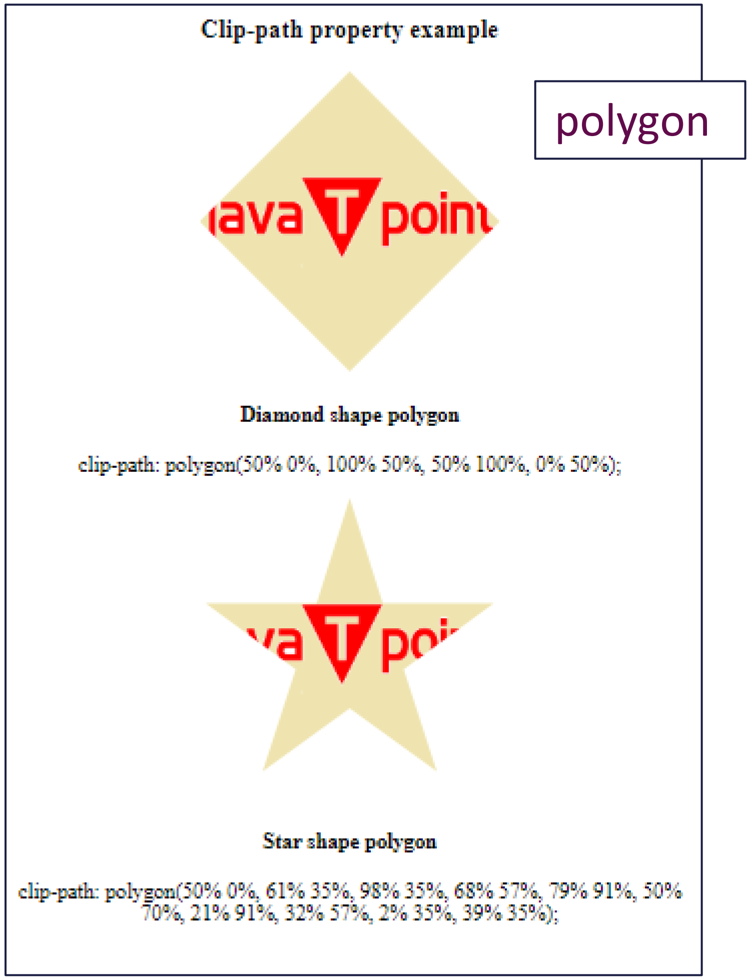
CSS clip-path example


Clip-path property example

circle

clip-path: circle(50%);

clip-path: circle(60% at 70% 20%);


Clip-path property example

polygon

Diamond shape polygon

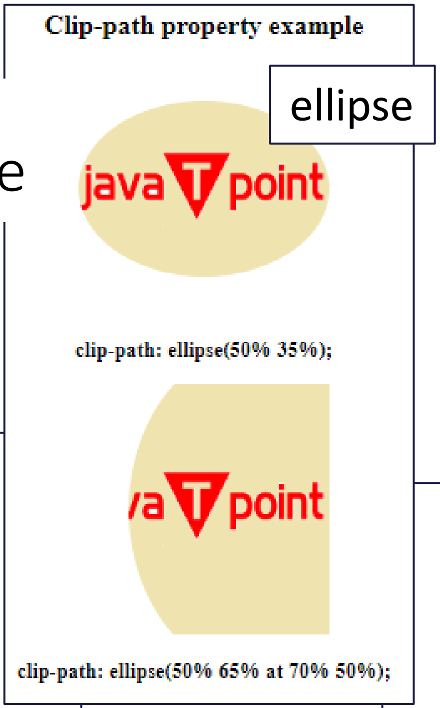
clip-path: polygon(50% 0%, 100% 50%, 50% 100%, 0% 50%);

Star shape polygon

clip-path: polygon(50% 0%, 61% 35%, 98% 35%, 68% 57%, 79% 91%, 50% 70%, 21% 91%, 32% 57%, 2% 35%, 39% 35%);

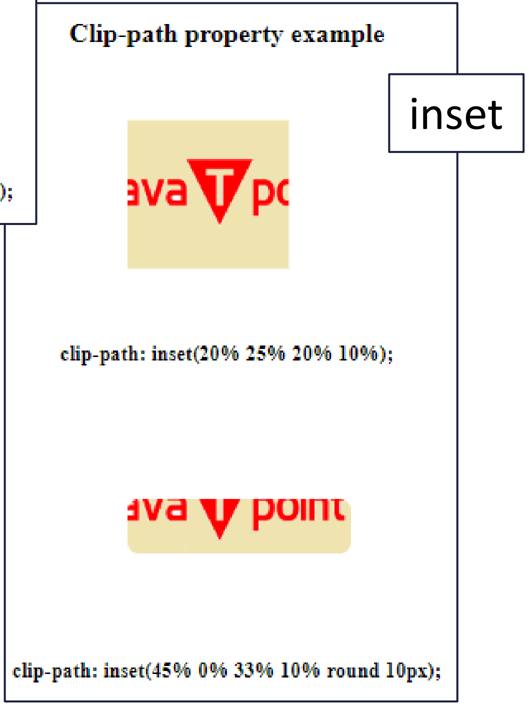
CSS clip-path example

```
<style>
img.example {   animation: anime 4s infinite;
  border: 5px dashed red; }

@keyframes anime {
  0% { clip-path: polygon(0 0, 100% 0, 100% 80%, 0% 70%);  }
  20% { clip-path: polygon(40% 0, 50% 0, 100% 100%, 0% 80%);  }
  40% { clip-path: polygon(0 0, 60% 72%, 100% 100%, 0 35%);  }
  60% { clip-path: polygon(70% 0, 20% 0, 100% 100%, 0% 80%);  }
  80% { clip-path: polygon(0 70%, 60% 0, 100% 32%, 0 40%);  }
  100% { clip-path: polygon(0 0, 60% 0, 100% 100%, 0% 30%);  }
}
 </style>
  <body>
..
    <h2> Animation in Clip-path property </h2>
          <img src="jtp.png" class="example">…
```



clip-path: ellipse(50% 35%);

clip-path: ellipse(50% 65% at 70% 50%);



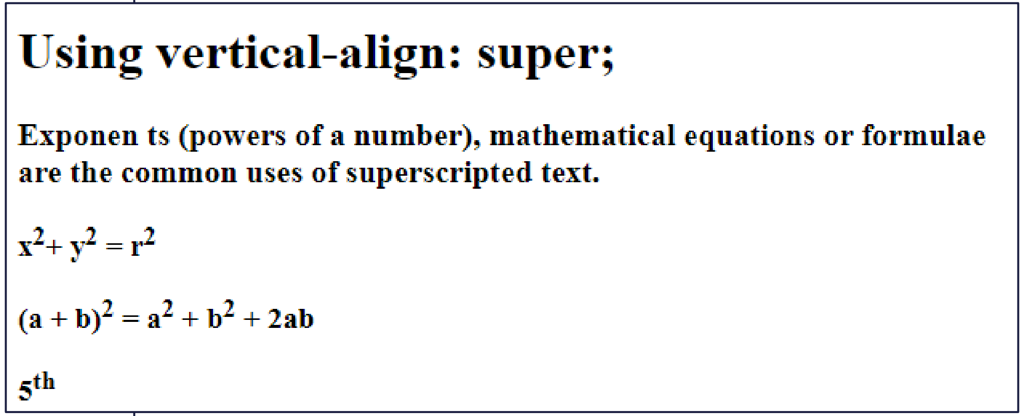
clip-path: inset(20% 25% 20% 10%);

clip-path: inset(45% 0% 33% 10% round 10px);

CSS Superscript and Subscript

- In CSS, the vertical-align property is used to specify the superscript and subscript of text using.
  - CSS property specifies the vertical alignment of the text
- Syntax
  - vertical-align: baseline | super | sub ;

```
……
#super{ vertical-align:super;
            font-size: medium;}
p{  font-weight: bold;
    font-size: 20px; }
</style>
<h1> Using vertical-align: super; </h1>
<p> Exponen ts (powers of a number), mathematical
equations or formulae are the common uses of
superscripted text. </p>
<h3>x<span id="super">2</span>+ y<span
id="super">2</span> = r<span id="super">2</span></h3>
<h3> (a + b)<span id="super">2</span> = a<span
id="super">2</span> + b<span id="super">2</span> + 2ab
</h3>
<h3>5<span id="super">th</span></h3>…
```



Using vertical-align: super;

Exponen ts (powers of a number), mathematical equations or formulae are the common uses of superscripted text.

$x^2 + y^2 = r^2$

$(a + b)^2 = a^2 + b^2 + 2ab$

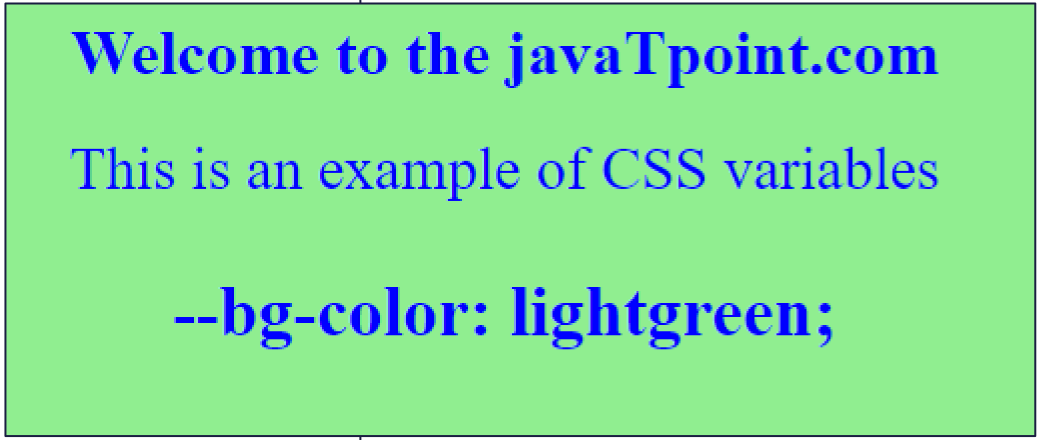
5th

CSS Variables

- The CSS variables are used to add the values of custom property to our web page that can be reused throughout the document
- These entities are set using the custom property notation and can be accessed by using the var() function
- The variables store the values and have a scope in which they can be used
- Syntax to define variable: element { --main-color: brown; }
- The names of the custom properties are case-sensitive
- The **var() function** in CSS is used to insert the custom property value. The name of the variable can be passed as the argument to the var() function
  - Syntax var(--custom-name, value)
- The var() function only allows two arguments that are defined as follows:
  - **--custom-name**: This parameter accepts the name of the custom property. It must begin with the two dashes (--). It is the required parameter
  - **value**: It is an optional parameter and accepts the fallback value. It is used as the substitution when the custom property is invalid

CSS Variables example

```
…….
       <style>
          :root {  --bg-color: lightgreen; }
          body { background-color: var(--bg-color);
           text-align: center;  }
          h1 {  color: var(--text-color, blue);  /* --text-color is not set,
so the fallback value 'blue' will be used */         }
          div {      color: var(--text-color, blue);
        font-size: 30px;     }
       </style>
   </head>

   <body>
      <h1>Welcome to the javaTpoint.com</h1>
      <div>
         This is an example of CSS variables
        <h3>--bg-color: lightgreen;</h3>…
```



Welcome to the javaTpoint.com

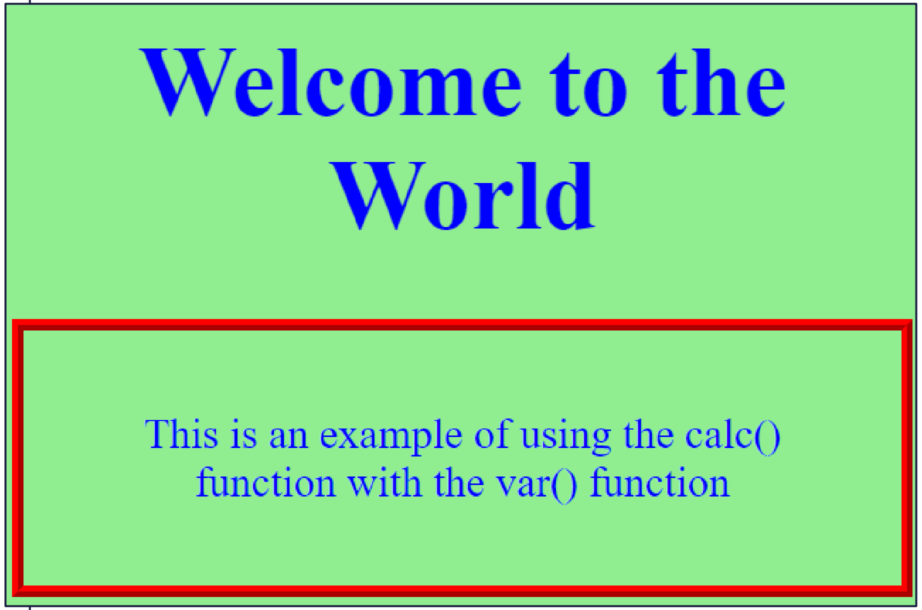
This is an example of CSS variables

--bg-color: lightgreen;

CSS Variables example

- Use of calc() with the var()

```
<style>
:root {
--bg-color: lightgreen;
--extra-padding: 1.2em;
--txt-size: 90px;
}
body { background-color: var(--bg-color); }
h1 {
color: var(--text-color, blue); /* --text-color is not set, so the
fallback value 'blue' will be used */
font-size: calc(var(--txt-size) - 20px);
}
div {
color: var(--text-color, blue);
font-size: 30px;
border: 8px ridge red;
padding: calc(var(--extra-padding) + 20px);
}
………………
<body>
<h1>Welcome to the World</h1>
<div> This is an example of using the calc() function with the
var() function </div>….
```



Welcome to the World

This is an example of using the calc() function with the var() function

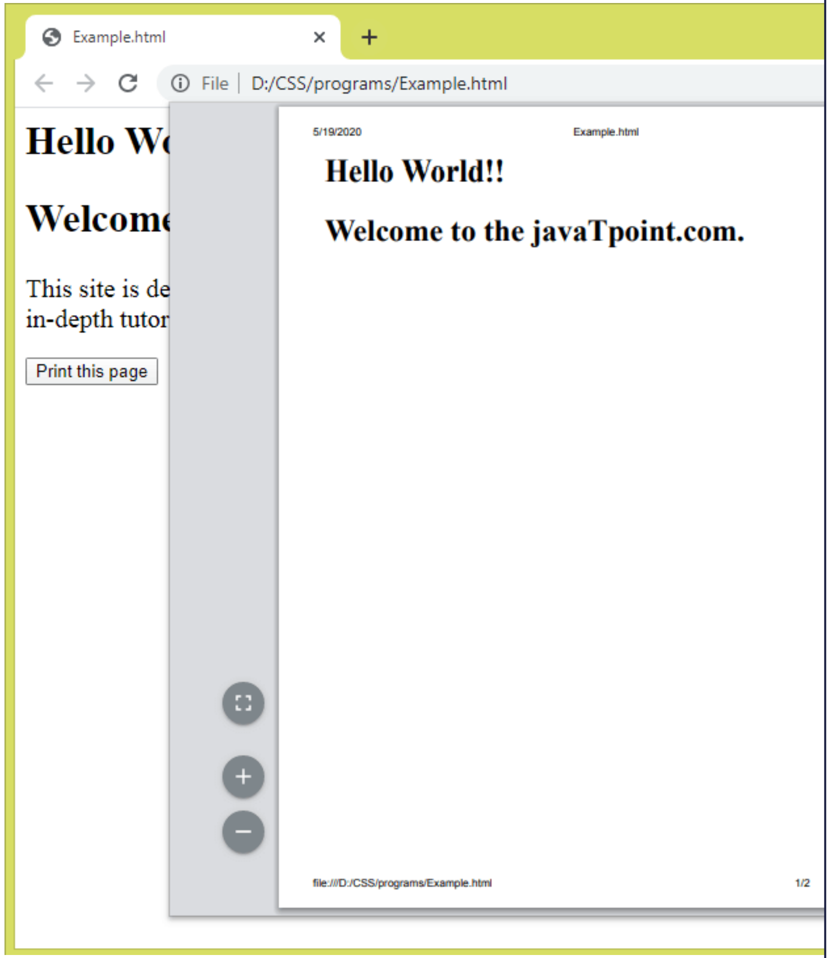
CSS page break

- **page-break-before property**: used to add the page break before the element, when printing the document
  - We cannot use this CSS property on absolutely positioned elements or an empty <div> element that does not generate a box.
- Syntax
  - page-break-before: auto | always | left | right | avoid | initial | inherit;

Values	Description
auto	It is the default value that inserts a page break before the element, if necessary
always	This value always forces a page break before the specified element
avoid	It is used to avoid a page break before the element whenever possible
left	This value forces either one or two page breaks before the element so that the next page will be depicted as the left-hand page
right	The **right** value forces either one or two page breaks before the element so that the next page will be depicted as the right-hand page
initial	It sets the property to its default value
inherit	If this value is specified, the corresponding element uses the computed value of its parent element **page-break-before** property

CSS page-break-before example

- Using the right value



Example.html

D:/CSS/programs/Example.html

Hello W

Welcome

This site is de
in-depth tutor

Print this page

5/19/2020 Example.html

Hello World!!

Welcome to the javaTpoint.com.

file:///D:/CSS/programs/Example.html 1/2

```
....
    <style type = "text/css">
      div{      font-size: 20px;
                page-break-before: right; }
    </style>
........
  <body>
    <div>
         <h2>Hello World!!</h2>
         <h2>Welcome to the javaTpoint.com.</h2>
    </div>
    <div>
This site is developed so that students may learn computer
science related technologies easily. The javaTpoint.com is
committed to providing easy and in-depth tutorials on
various technologies. No one is perfect in this world, and
nothing is eternally best. But we can try to be better.
</div>    <br>
    <button onclick = "func()">Print this page</button>
     <script>
      function func() {  window.print();  }
    </script>....
```

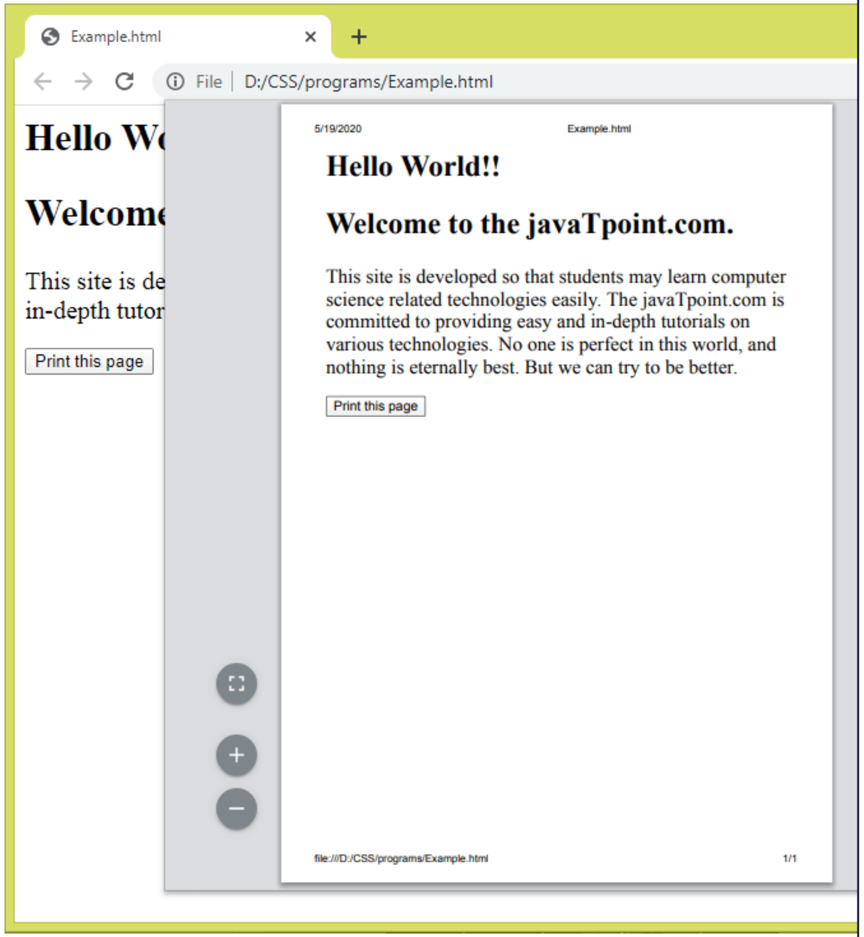
CSS page-break-inside property

- **page-break-inside** property is used to specify the page break inside the element, when printing the document
- Syntax
  - page-break-inside: auto | avoid | initial | inherit;

Values	Description
auto	It is the default value that inserts a page break inside the element, if necessary.
avoid	It is used to avoid a page break inside the element whenever possible.
initial	It sets the property to its default value.
inherit	If this value is specified, the corresponding element uses the computed value of its parent element **page-break-inside** property.

CSS page-break-inside example

- Using the avoid value


Example.html

File | D:/CSS/programs/Example.html

5/19/2020 Example.html

Hello World!!

Welcome to the javaTpoint.com.

This site is developed so that students may learn computer science related technologies easily. The javaTpoint.com is committed to providing easy and in-depth tutorials on various technologies. No one is perfect in this world, and nothing is eternally best. But we can try to be better.

Print this page

file:///D:/CSS/programs/Example.html 1/1

```
….
    <style type = "text/css">
      div{
                font-size: 20px;
                page-break-inside: avoid;
                }
    </style>
  </head>
  <body>
    <div>
         <h2>Hello World!!</h2>
         <h2>Welcome to the javaTpoint.com.</h2>
    </div>
    <div>
This site is developed so that students may learn computer
science related technologies easily. The javaTpoint.com is
committed to ……in this world, and nothing is eternally best.
But we can try to be better.
</div>
    <br>
    <button onclick = "func()">Print this page</button>

    <script>  function func() {  window.print();  }  </script>
```

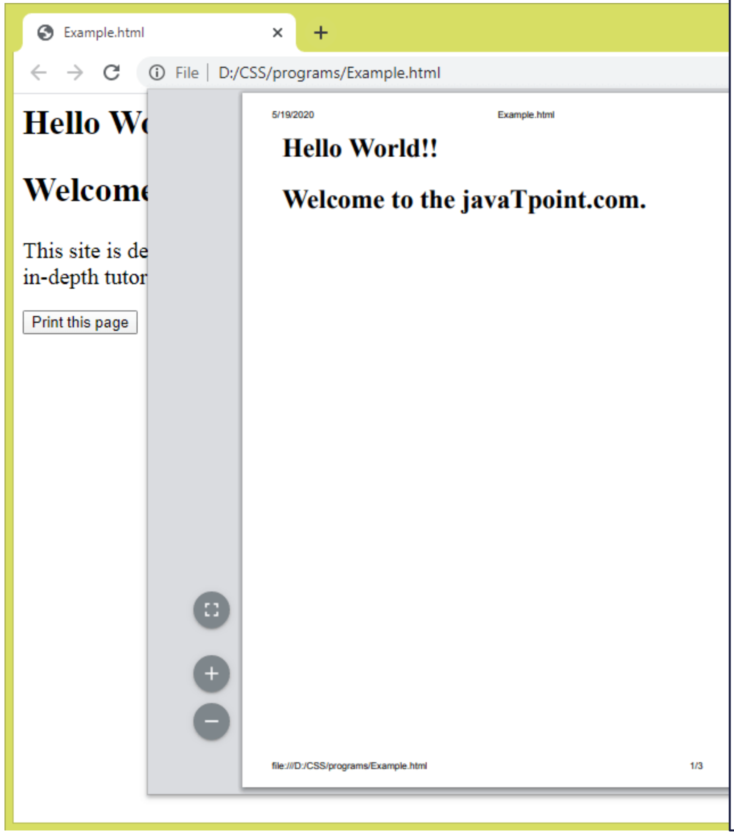
CSS page-break-after property

- **page-break-after** property is used to adjust the page break after the element when printing the document
- Syntax
  - page-break-after: auto | always | left | right | avoid | initial | inherit;

Values	Description
auto	It is the default value that inserts a page break after the element, if necessary.
Always	It always forces a page break after the specified element.
avoid	It is used to avoid a page break after the element whenever possible.
left	It forces either one or two page breaks after the specified element so that the next page will be depicted as the left-hand page.
right	It forces either one or two page breaks after the specified element so that the next page will be depicted as the right-hand page.
Initial	It sets the property to its default value.
Inherit	If this value is specified, the corresponding element uses the computed value of its parent element.

CSS page-break-after example

- Using the always value



Example.html

File | D:/CSS/programs/Example.html

Hello W

Welcome

This site is de
in-depth tutor

Print this page

5/19/2020 Example.html

Hello World!!

Welcome to the javaTpoint.com.

file:///D:/CSS/programs/Example.html 1/3

```
....
     <style type = "text/css">
       div{   font-size: 20px;
                 page-break-after: always; }
     </style>
.....
     <div>
          <h2>Hello World!!</h2>
          <h2>Welcome to the javaTpoint.com.</h2>
     </div>
     <div>
This site is developed so that students may learn
computer science .......this world, and nothing is
eternally best. But we can try to be better.
</div>
     <br>
     <button onclick = "func()">Print this page</button>
     <script>   function func() { window.print();  }
</script>
```

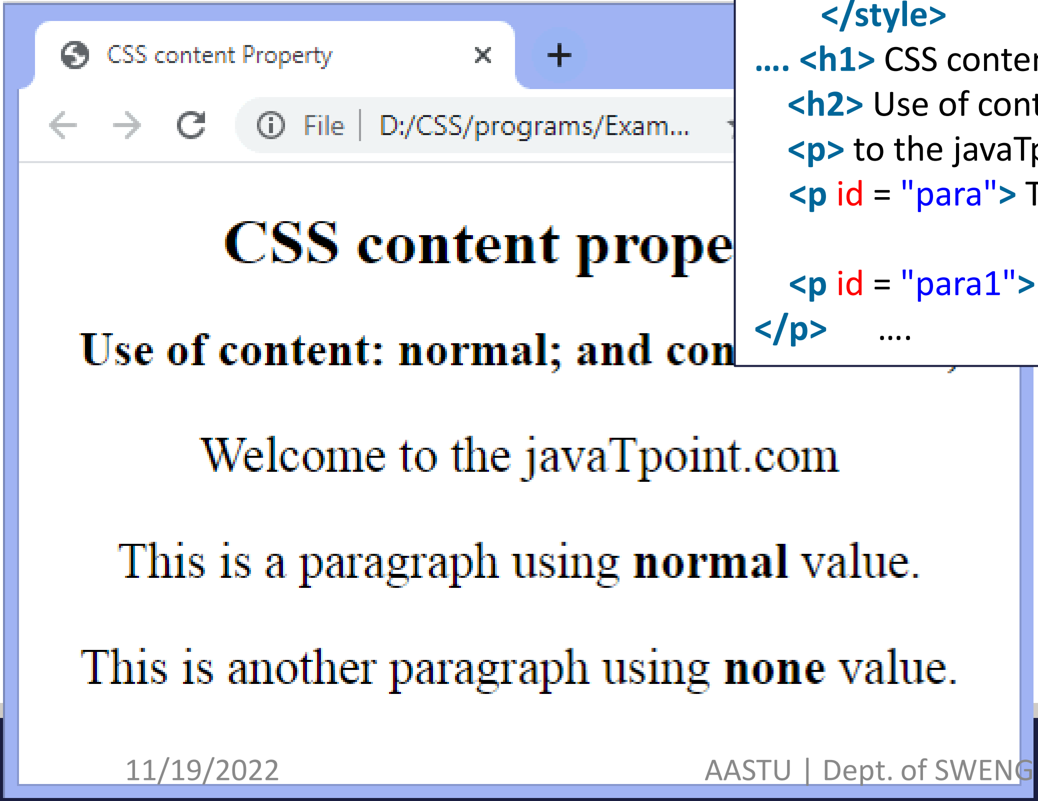
CSS content property

- This CSS property generates dynamic content. It can be used with the pseudo-elements ::**before** and ::**after**
- Syntax
  - content: normal | none | counter | string | attr | open-quote | close-quote | no-close-quote | no-open-quote | url | initial | inherit;

Values	Description
normal	It is used to set the default value
none	This value does not set the content.
counter	It sets the content as the counter. It is generally a number. It is displayed by using the **counter()** or **counters()** function.
string	It is used to set any string value. It should always be enclosed within quotation marks. It generates any string after or before the HTML element.
attr	It inserts the value of the specified attribute after or before the element. If the selector does not have a particular attribute, then an empty string will be inserted.
open-quote	It is used to insert the opening quotation mark, or it sets the content to be an opening quote.
close-quote	It is used to insert the closing quotation mark, or it sets the content to be a closing quote.
no-close-quote	If the closing quotes are specified, then it is used to remove the closing quote from the content.
no-open-quote	If the opening quotes are specified, then it is used to remove the opening quote from the content.
url	It is used to set the content to some media, which could be an image, video, audio, and many more.
Initial	It is used to set the property to its default value.
Inherit	It inherits the property from its parent element.

CSS content property example

```
…..
   <style>
   body{  text-align: center;  }
   p{  font-size: 25px;   }
   p::before {  content: "Welcome ";   }
    #para::before {   content: normal;    }
    #para1::before {     content: none;     }
      </style>
…. <h1> CSS content property </h1>
   <h2> Use of content: normal; and content: none; </h2>
   <p> to the javaTpoint.com </p>
   <p id = "para"> This is a paragraph using <b>normal</b> value. </p>

   <p id = "para1"> This is another paragraph using <b>none</b> value.
</p>     ….
```



CSS content prope

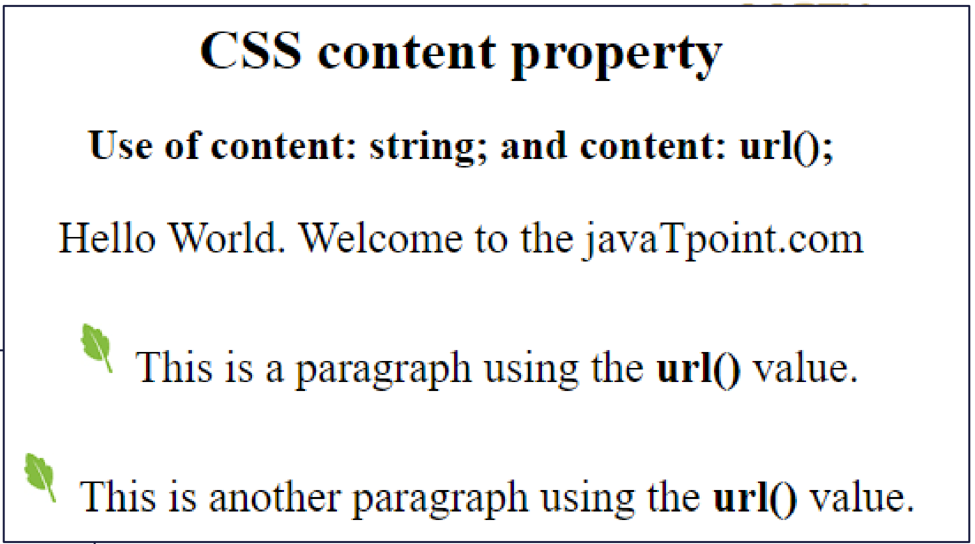
Use of content: normal; and con

Welcome to the javaTpoint.com

This is a paragraph using **normal** value.

This is another paragraph using **none** value.

CSS content example using string and url value



CSS content property

Use of content: string; and content: url();

Hello World. Welcome to the javaTpoint.com

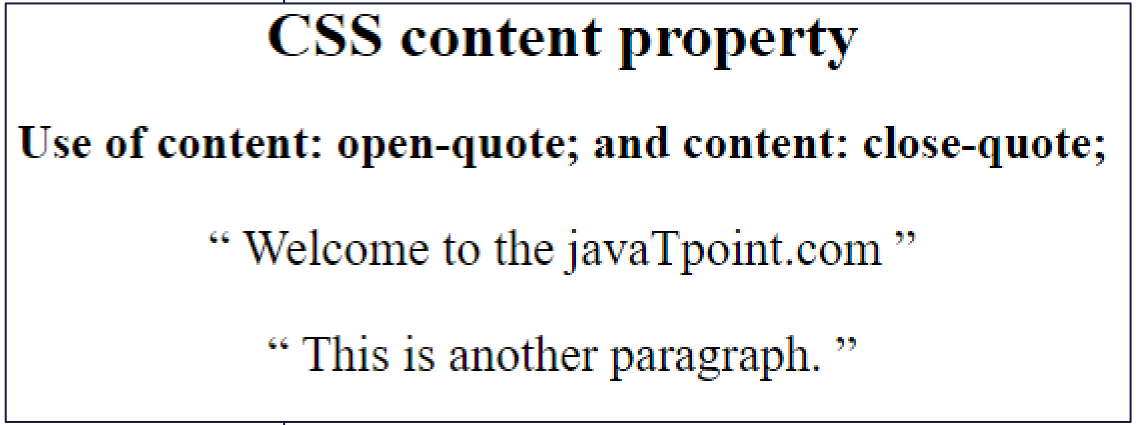
This is a paragraph using the **url()** value.

This is another paragraph using the **url()** value.

```
…
p{ font-size: 25px;       }
p::before {  content: "Hello World. Welcome ";  }
 #para::before {  content: url("img.png");  }
 #para1::before {  content: url("img.png"); }
</style>
….
   <h1> CSS content property </h1>
   <h2> Use of content: string; and content: url();
</h2>
   <p> to the javaTpoint.com </p>
   <p id = "para"> This is a paragraph using the
<b>url()</b> value. </p>
   <p id = "para1"> This is another paragraph using
the <b>url()</b> value. </p>        ..
```

CSS content example using open-quote and close-quote value

```
....
body{ text-align: center; }
p{ font-size: 25px;}
p::before {content: open-quote;}
p::after { content: close-quote }
 </style>
</head>
..
   <h1> CSS content property </h1>
   <h2> Use of content: open-quote; and
content: close-quote; </h2>
   <p> Welcome to the javaTpoint.com </p>
   <p> This is another paragraph. </p>
</body>
</html>
```

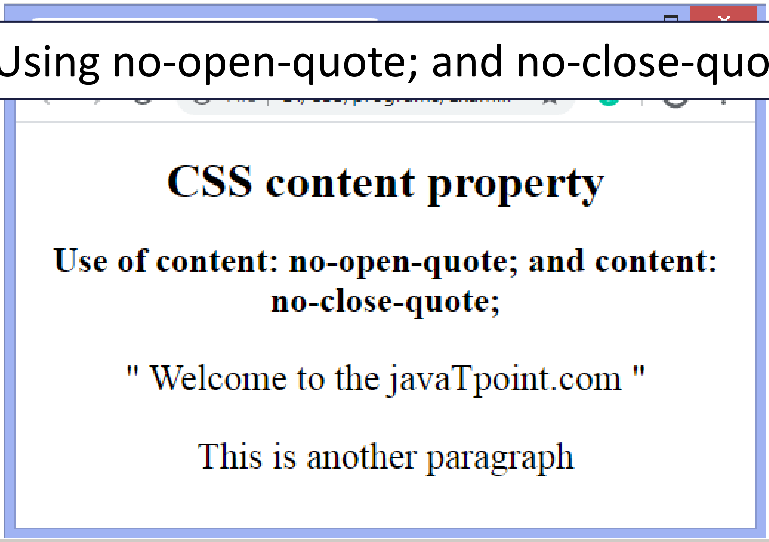

CSS content property

Use of content: open-quote; and content: close-quote;

“ Welcome to the javaTpoint.com ”

“ This is another paragraph. ”

Using no-open-quote; and no-close-quote


CSS content property

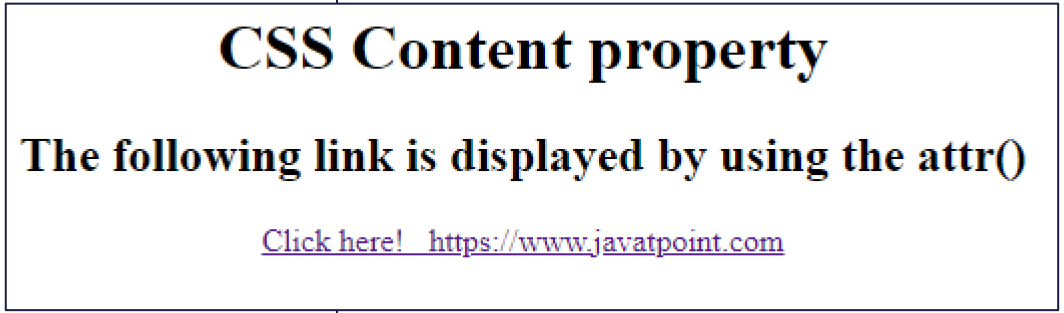
Use of content: no-open-quote; and content: no-close-quote;

" Welcome to the javaTpoint.com "

This is another paragraph

CSS content example using attr()

```
….
  <style>
       body{ text-align: center; }
      a::after { content: attr(href); }
  </style>
……
<h1> CSS Content property </h1>
<h2> The following link is displayed by using the
<b>attr()</b> </h2>
   <a href= https://www.javatpoint.com>Click
here!  </a> …
```


CSS Content property

The following link is displayed by using the attr()

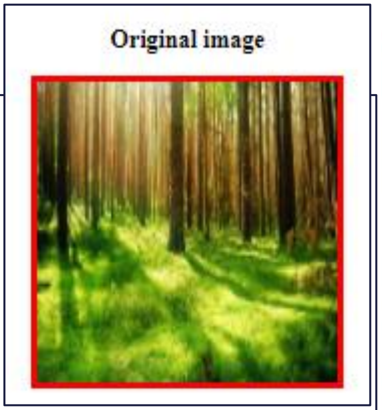
Click here! https://www.javatpoint.com

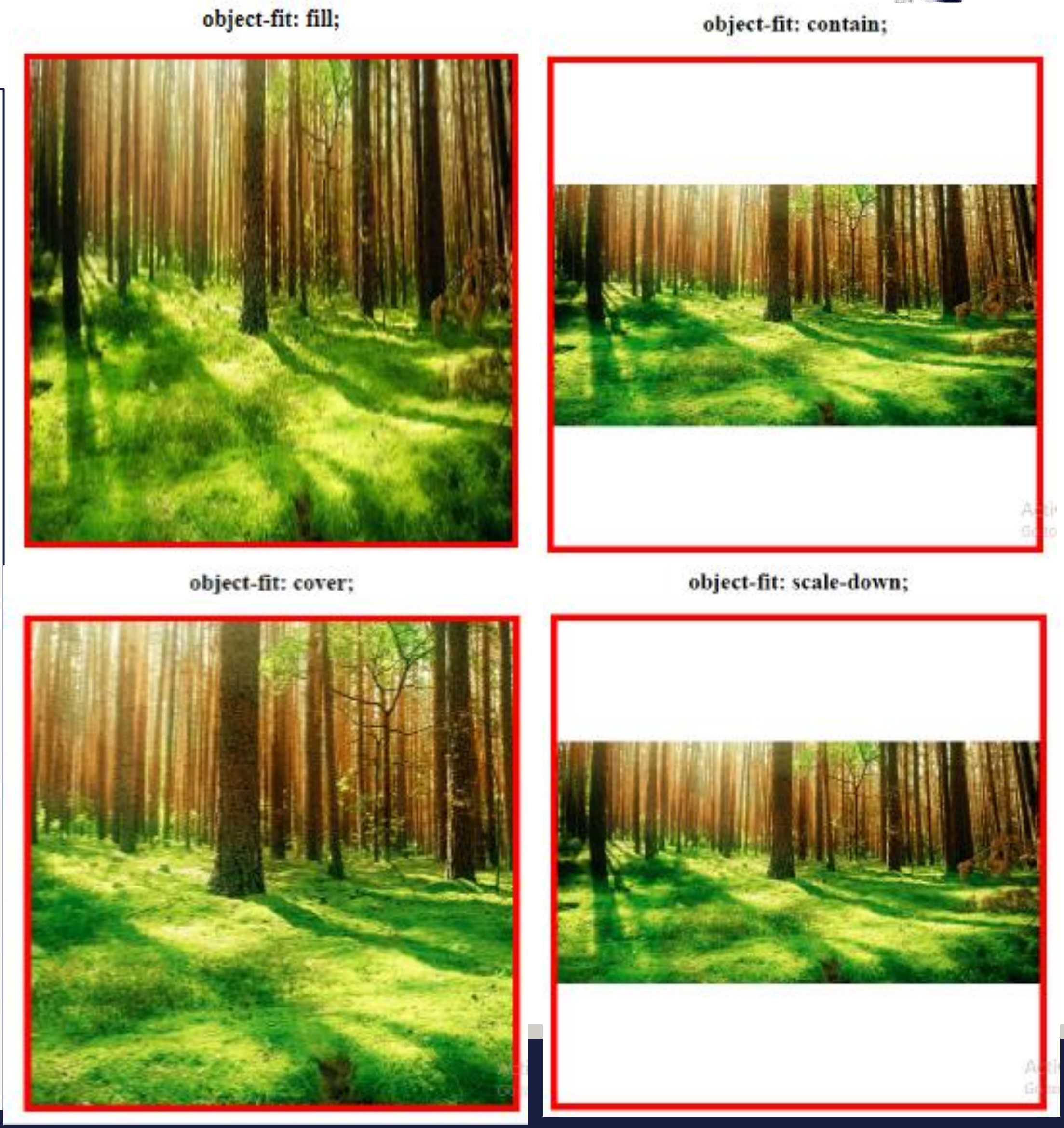
CSS object-fit property

- This CSS property specifies how a video or an image is resized to fit its content box
- It defines how an element fits into the container with an established width and height
- Syntax
  - object-fit: fill | contain | cover | none | scale-down | initial | inherit;
- **fill**: It is the default value. Using this value, the entire object fills in the container. The replaced content is sized to fill the content box. If the aspect ratio of the object doesn't match the aspect ratio of the box, the object will be squished or stretched to fit in the box
- **contain**: It resizes the content of an element to stay contained inside the container. It fits the image in the width and height of the box while maintaining the aspect ratio of the image. The replaced content is scaled for maintaining the aspect ratio while fitting within the content box of the element.
- **cover**: It resizes the content of an element to cover its container. It fills the entire box with the image. If the aspect ratio of the object is not matched with the aspect ratio of the box, it clips the object to fit
- **none**: It does not resize the replaced content. The image retains its original size and ignores the height and width of the parent
- **scale-down**: It sized the content as none or as contain. It results in the smallest object size. It finds the smallest concrete object size by comparing the none and contain

CSS object-fit property example

```
..
#img1{ width: 300px;
height: 300px;
border: 7px solid red;
#obj { width: 500px;
height: 500px;
object-fit: fill;
border: 7px solid red; }
#left{ float: left;
text-align: center;
padding-left: 200px; }
#center{ float: center;
text-align: center; }
.....
<h1> Example of Object-fit property </h1>
<div id = "left">
<h2> Original image </h2>
<img id = "img1" src = "forest.jpg">
</div>
<div id= "center">
<h2> object-fit: fill; </h2>
<img id = "obj" src="forest.jpg" width="300"
height="300"</...
```





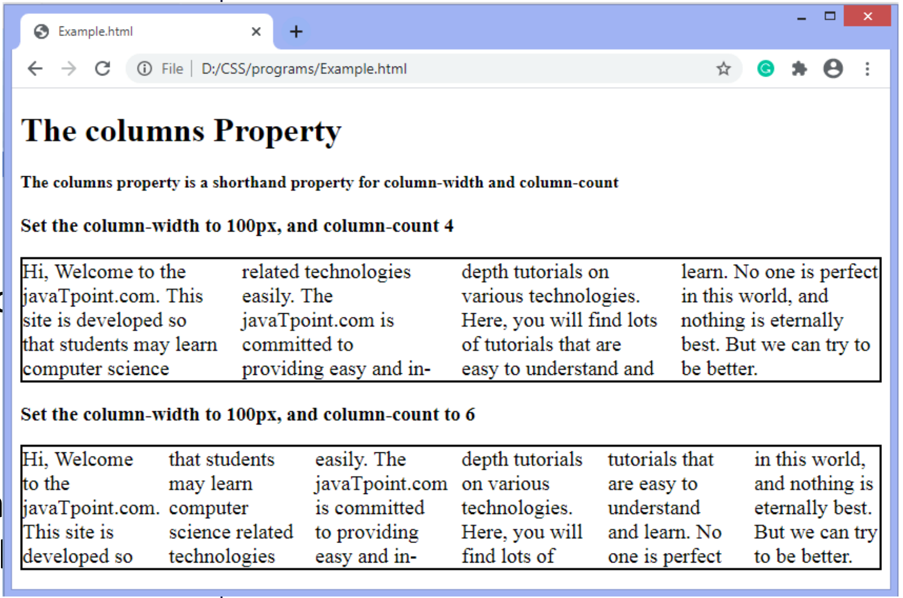
CSS columns

- The columns property in CSS sets the number and width of columns in a single declaration.
- It is used to set both column-count and column-width properties at the same time
- Syntax
  - columns: auto | column-width column-count| initial | inherit;

Value	Description
auto	It is the default value which sets the values of **column-count** and **column-width** to the default browser values.
column-width	It is used to set the minimum width for columns. However, the actual width of the column may be narrower or wider based on the available space.
column-count	It specifies the maximum number of columns.
Initial	It is used to set the property to its default value.
Inherit	It inherits the property from its parent element.

CSS columns example

```
…..
.div1 {  columns: 100px 4;
  border: solid 2px black;
  font-size: 20px;  }
.div2 {  columns: 100px 6;
  border: solid 2px black;
  font-size: 20px;  }
</style>
….
<h1> The columns Property </h1>
<h4> The columns …-count </h4>
<h3>   Set   the   column-width   to   100px,   and
count 4 </h3>
<div class="div1">
Hi,                 Welcome…..easy                 and
depth ….ots of tutorials that are easy to understand a
No one is perfect in this world, and nothing is eternal
 we can try to be better.
</div>

<h3>Set   the   column-width   to   100px,   and   column-
count to 6 </h3>
<div class="div2">
Hi, Welcome to the... This site is ..to providing easy and in-
depth tutorials on …best. But we can try to be better.
</div>  ….
```



Example.html

D:/CSS/programs/Example.html

The columns Property

The columns property is a shorthand property for column-width and column-count

Set the column-width to 100px, and column-count 4

```
Hi, Welcome to the        related technologies       depth tutorials on         learn. No one is perfect
javaTpoint.com. This      easily. The                various technologies.      in this world, and
site is developed so      javaTpoint.com is          Here, you will find lots   nothing is eternally
that students may learn   committed to               of tutorials that are      best. But we can try to
computer science          providing easy and in-     easy to understand and     be better.
```

Set the column-width to 100px, and column-count to 6

```
Hi, Welcome      that students    easily. The       depth tutorials   tutorials that   in this world,
to the           may learn        javaTpoint.com    on various        are easy to      and nothing is
javaTpoint.com.  computer         is committed      technologies.     understand       eternally best.
This site is     science related  to providing      Here, you will    and learn. No    But we can try
developed so     technologies     easy and in-      find lots of      one is perfect   to be better.
```

CSS pointer-events property

- This CSS property specifies whether or not an element shows some action when the pointer event is triggered upon it
- The pointer-events are triggered by touch, stylus, mouse click, and many more
- Syntax
  - pointer-events: none | auto |initial | inherit;

```
p{
        font-size: 20px;
        pointer-events: none;
}
```

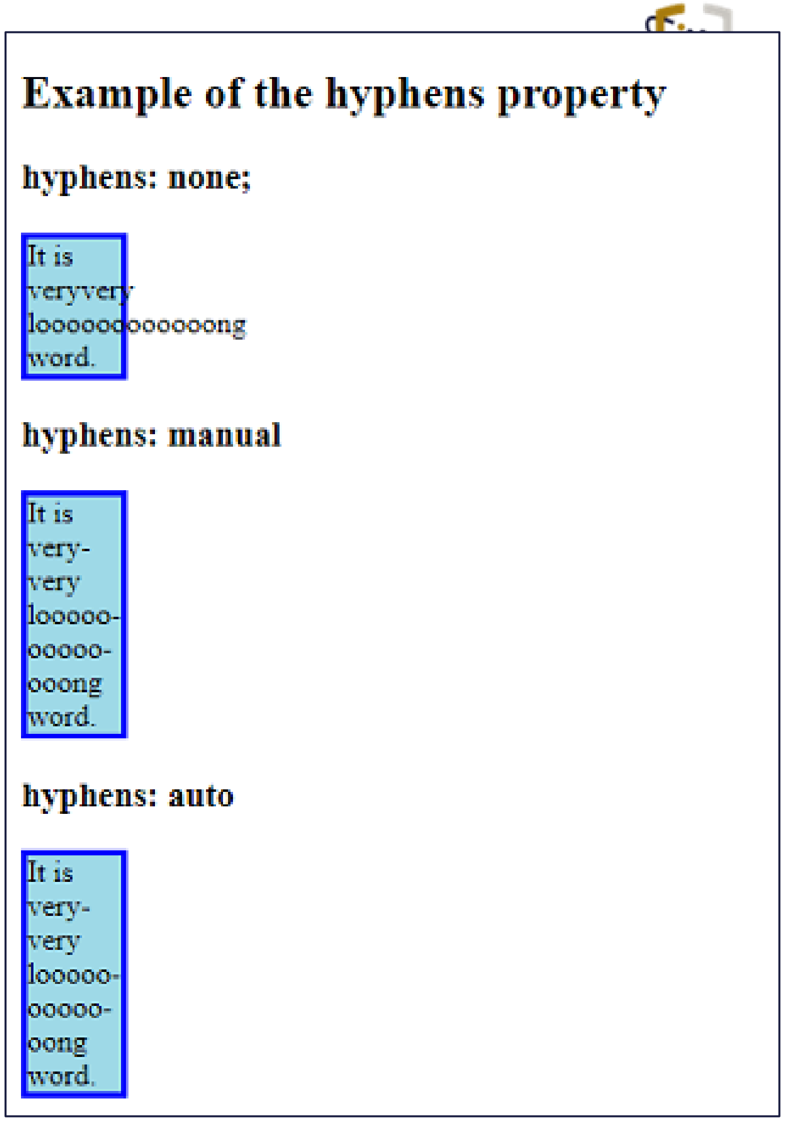
```
p{
        font-size: 20px;
        pointer-events: auto;
}
```

CSS hyphens property

- This CSS property defines how the word is hyphenated if it is too long or when the text wraps across multiple lines
- Syntax
  - hyphens: none | manual | auto | initial | inherit;
- **none**: This value does not hyphenate the words. It never hyphenates the words at line breaks or even if the word is too long.
- **manual**: It is the default value that hyphenates the word only when the characters in the word suggest hyphenation opportunities
  - The two Unicode characters are defined below, which can be used manually to specify the possible line breakpoints in the text
    - **U+2010 (HYPHEN)** - It is the 'hard' hyphen character that specifies the visible line-break opportunity. The hyphen is rendered, even if the line is not broken at that point.
    - **U+00AD (SHY)** - It is an invisible 'soft' hyphen. It is not visibly rendered; instead, it spots the place where the word should be required to break. In html, for a soft hyphen, we can use &shy;
- **auto**: In this value, the algorithm decides where the words are hyphenated

CSS hyphens example

```
….
<style>
div {  width: 50px;
border: 3px solid blue;
background-color: lightblue;  }
 .none{  hyphens: none;  }
 .manual{  hyphens: manual;  }
 .auto{  hyphens: auto;  }
</style>
…..
<h2> Example of the hyphens property </h2>

<h3> hyphens: none; </h3>
<div class="none"> It is veryvery looooo-
oooooong word.  </div>
<h3>hyphens: manual</h3>
<div class="manual">  It is veryvery looooo-
ooooooong word.  </div>
<h3>hyphens: auto</h3>
<div class="auto">
It is very-very looooo-ooooo-oong word.
</div>  …..
```



Example of the hyphens property

hyphens: none;

It is
veryvery
loooooooooooong
word.

hyphens: manual

It is
very-
very
looooo-
ooooo-
ooong
word.

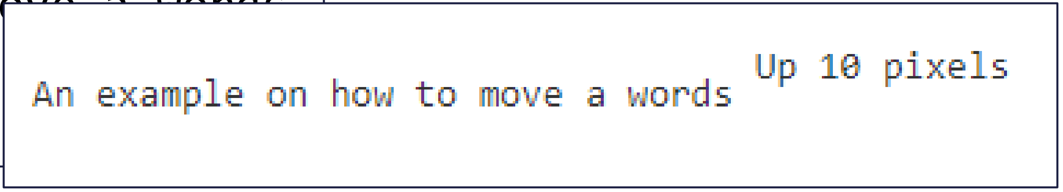
hyphens: auto

It is
very-
very
looooo-
ooooo-
oong
word.

CSS Position Offset Properties

- The Offset CSS Properties of a box are **top**, **left**, **light**, **bottom**
- They are applied to the containing box in the following positioning model in order to position a box
  - relative, absolute or fixed
- They are used then only when the box has one of the following position value:
  - relative, absolute or fixed
- Example

```
.up {
      position:relative;
      bottom:10px;
}
…….
An example on how to m
<span class="up">Up 10
pixels</span>
```



```
An example on how to move a words  Up 10 pixels
```

CSS Max-width, Min-width, Max-height, and Min-height

- The **max-width** property lets you specify an element's box maximum width
  - This means that element's box can increase in width until it reaches a specific absolute or relative unit, at which point it should fix its width to that unit
- The **min-width** property specifies the minimum width

Example using max-width

```
//The card can not get larger
than 350px.
.card{ margin:0 auto;
       padding:1.5em;
       width:80%;
       max-width:350px;
       height:50%;
       background: #FFFFFF;
       box-shadow: 0px 0px 5px
       rgba(0, 0, 0, 0.3);
       border-radius:4px;
       overflow:hidden;
}
```

Example using min-width

```
//Here the card element can
not get smaller than 250px
.card{ margin:0 auto;
 padding:1.5em;
 width:80%;
 max-width:350px;
 //here we set the min-width
property
 min-width : 250px;
 height:50%;
 background: #FFFFFF;
 box-shadow: 0px 0px 5px
rgba(0, 0, 0, 0.3);
 border-radius:4px;
 overflow:hidden;
}
```

CSS Max-width, Min-width, Max-height, and Min-height

- **max-height:** this property operates similarly to the max-width property, but it affects the height instead of the width
- **min-height** property provides a minimum height for an element

Example using max-height

```
// it fixes the height of an element to a
specified unit, effectively stopping it from
increasing in height
.card{
 margin:0 auto;
 padding:1.5em;
 width:80%;
 max-width:350px;
 height:70%;
 //here we set the max-height for the card.
 max-height: 400px;
 background: #FFFFFF;
 box-shadow: 0px 0px 5px rgba(0, 0, 0, 0.3);
 border-radius:4px;
 overflow:hidden;
}
```

Example using min-height

```
.element{
 height:40vh;
 min-height:200px;
 }
```

CSS cubic-bezier()

- It is an inbuilt function in CSS that defines a Cubic Bezier curve
- This CSS function is the transition timing function, which is used for the smooth and custom transitions
- It is defined by the four points (that are P0, P1, P2, and P3)
    - The points P0 and P3 (called as anchors) are the starting and the ending points
    - The points P1 and P2 (called as handles) are the middle points
- the points P0 and P3 are always in the same spot. P0 is at (0,0) that represents the initial state and initial time, and P3 is at (1,1), which represents the final state and the final time
- Syntax
    - cubic-bezier(x1, y1, x2, y2)

CSS cubic-bezier() example

```
…
.jtp {
        width: 200px;
        height: 50px;
        background: blue;
        transition: width 3s;
        animation-timing-function: cubic-
        bezier(.59,1.01,0,.01)
}
div:hover { width:300px; }
p{ font-size: 20px;
color: darkviolet; }
….
<h1> The cubic-bezier() Function </h1>
<p> Move your mouse over the below div element,
to see the transition effect. </p>
<div class = "jtp">……
```

CSS quotes

- The quotes property in CSS specifies the type of quotation mark for the quotation used in the sentence
- Syntax
  - quotes: none | string | initial;
- **none**: It is the default value that specifies that the open-quote and close-quote values of the content property do not generate any quotation marks.
- **string**: This value specifies the type of quotation mark to be used in sentence. The first pair represents the outer-level quotation; the second pair indicates the first nested level, the third pair indicates the third level, and so on

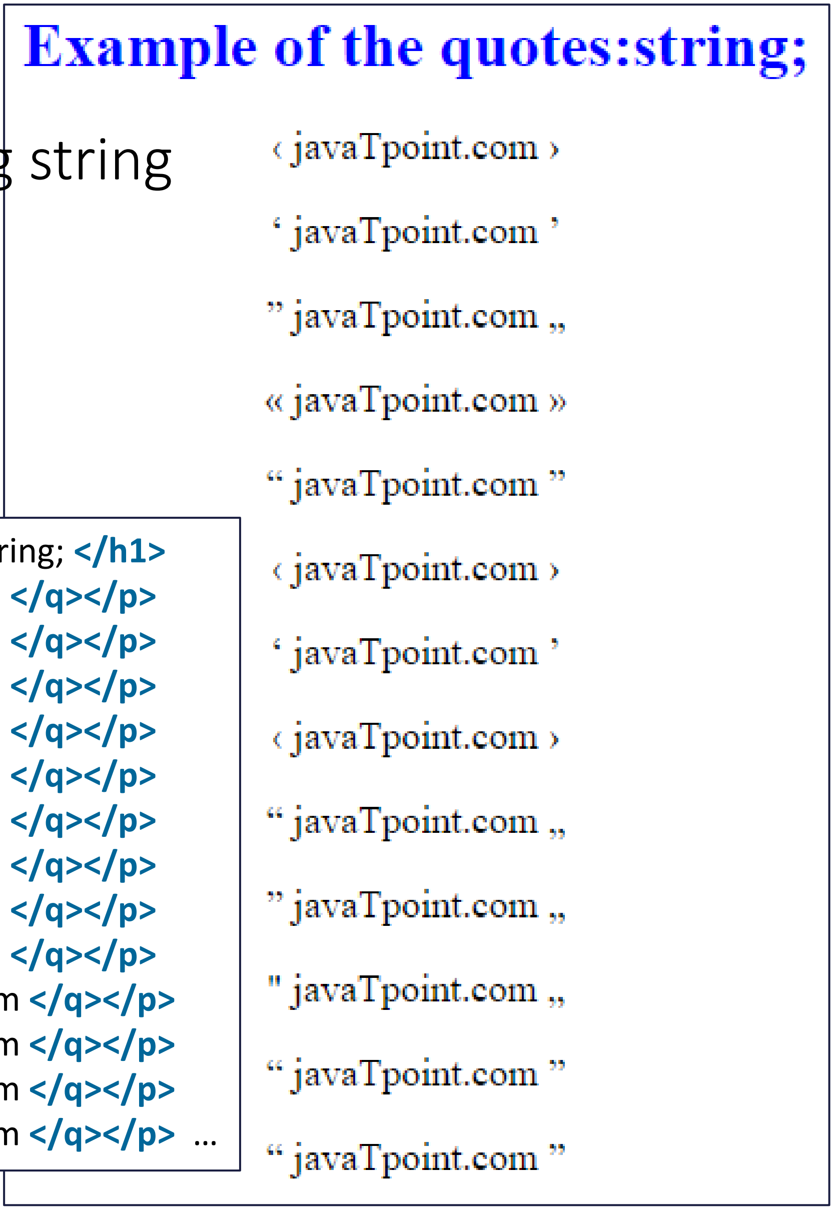
	Description	**Entity number**
"	double quote	\0022
'	single quote	\0027
„	double quote (double low-9)	\201E
«	double, left angle quote	\00AB
»	double, right angle quote	\00BB

	Description	**Entity number**
‹	single, left angle quote	\2039
›	single, right angle quote	\203A
'	left quote (single high-6)	\2018
'	right quote (single high-9)	\2019
“	left quote (double high-6)	\201C
”	right quote (double high-9)	\201D

CSS quotes example using string

```
…
body{  text-align: center;  }
h1{  color: blue;  }
p{  font-size: 20px;  }
#j1 {  quotes: '‹' '›';  }
#j2 {  quotes: '‘' '’' ;  }
#j3 {  quotes: '”' '„';  }
#j4 {  quotes: '«' '»';  }
#j5 {  quotes: '“' '”';  }
#j6 {  quotes: '‹' '›' '«' '»' ;  }
#j7 {  quotes: '\2018' '\2019';  }
#j8 {  quotes: '\2039' '\203A';  }
#j9 {  quotes: '\201C' '\201E';  }
#j10 {  quotes: '\201D' '\201E';  }
#j11 {  quotes: '\0022' '\201E';  }
#j12 {  quotes: '\201C' '\201D';  }
#j13 {  quotes: initial;  }
…………
```

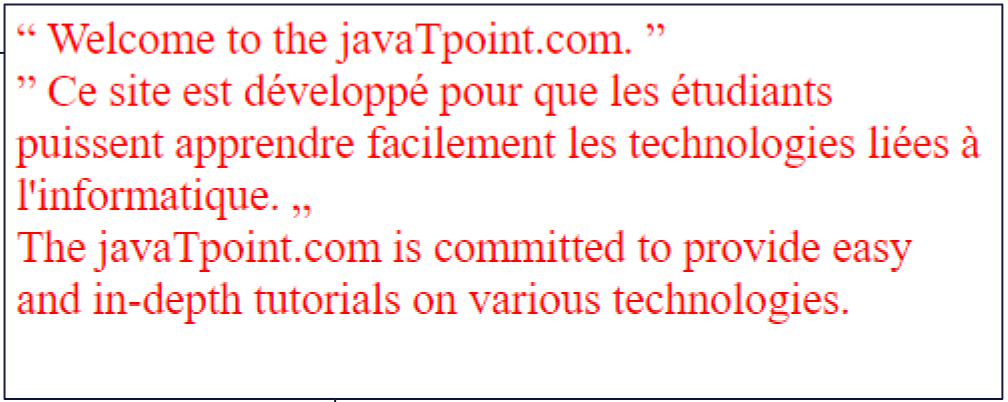
```
<h1> Example of the quotes:string; </h1>
<p><q id="j1"> javaTpoint.com </q></p>
<p><q id="j2"> javaTpoint.com </q></p>
<p><q id="j3"> javaTpoint.com </q></p>
<p><q id="j4"> javaTpoint.com </q></p>
<p><q id="j5"> javaTpoint.com </q></p>
<p><q id="j6"> javaTpoint.com </q></p>
<p><q id="j7"> javaTpoint.com </q></p>
<p><q id="j8"> javaTpoint.com </q></p>
<p><q id="j9"> javaTpoint.com </q></p>
<p><q id="j10"> javaTpoint.com </q></p>
<p><q id="j11"> javaTpoint.com </q></p>
<p><q id="j12"> javaTpoint.com </q></p>
<p><q id="j13"> javaTpoint.com </q></p>  …
```



```
Example of the quotes:string;

‹ javaTpoint.com ›
‘ javaTpoint.com ’
” javaTpoint.com „
« javaTpoint.com »
“ javaTpoint.com ”
‹ javaTpoint.com ›
‘ javaTpoint.com ’
‹ javaTpoint.com ›
“ javaTpoint.com „
” javaTpoint.com „
" javaTpoint.com „
“ javaTpoint.com ”
“ javaTpoint.com ”
```

CSS quotes example using :lang pseudo-class

```
.....
p{
font-size: 25px;
color: red;
}
:lang(en) { quotes: '“' '”'; }
:lang(fr) { quotes: '\201D' '\201E' ; }
</style>
.....
<p><q lang = "en"> Welcome to the javaTpoint.com. </q>
<br>
<q lang = "fr"> Ce site est développé pour que les étudiants
puissent apprendre facilement les technologies liées à
l'informatique.</q><br>
The javaTpoint.com is committed to provide easy and in-
depth tutorials on various technologies. </q></p>
</body>
</html>
```



“ Welcome to the javaTpoint.com. ”
” Ce site est développé pour que les étudiants puissent apprendre facilement les technologies liées à l'informatique. „
The javaTpoint.com is committed to provide easy and in-depth tutorials on various technologies.

CSS transform-origin property

- This CSS property is used to change the position of transformed elements
- The transform-origin property must be used with the transform property
- The 2d transformation can change the x-axis and y-axis of the element, whereas the 3D transformation can change the z-axis along with the x-axis and y-axis.
- This property can be specified by using one, two, or three values
  - The first value affects the horizontal position
  - The second value affects the vertical position, and
  - The third value indicates the position of the z-axis. The third value can also work on 3D transformations and cannot be defined using a percentage
- If we specify only one value, the value must be a length or percentage, or one of the keyword values left, center, right, top, and bottom.
- If we specify two values, the first value must be a length or percentage or the keyword values left, right, and center. The second value must be a length or percentage or one of the keyword values center, top, and bottom
- When we specify three values, then the first two values are same as the two-value syntax, but the third value represents the z-offset, so it must be a length
- The default value of the transform-origin property is 50% 50%, which represents the center of the element

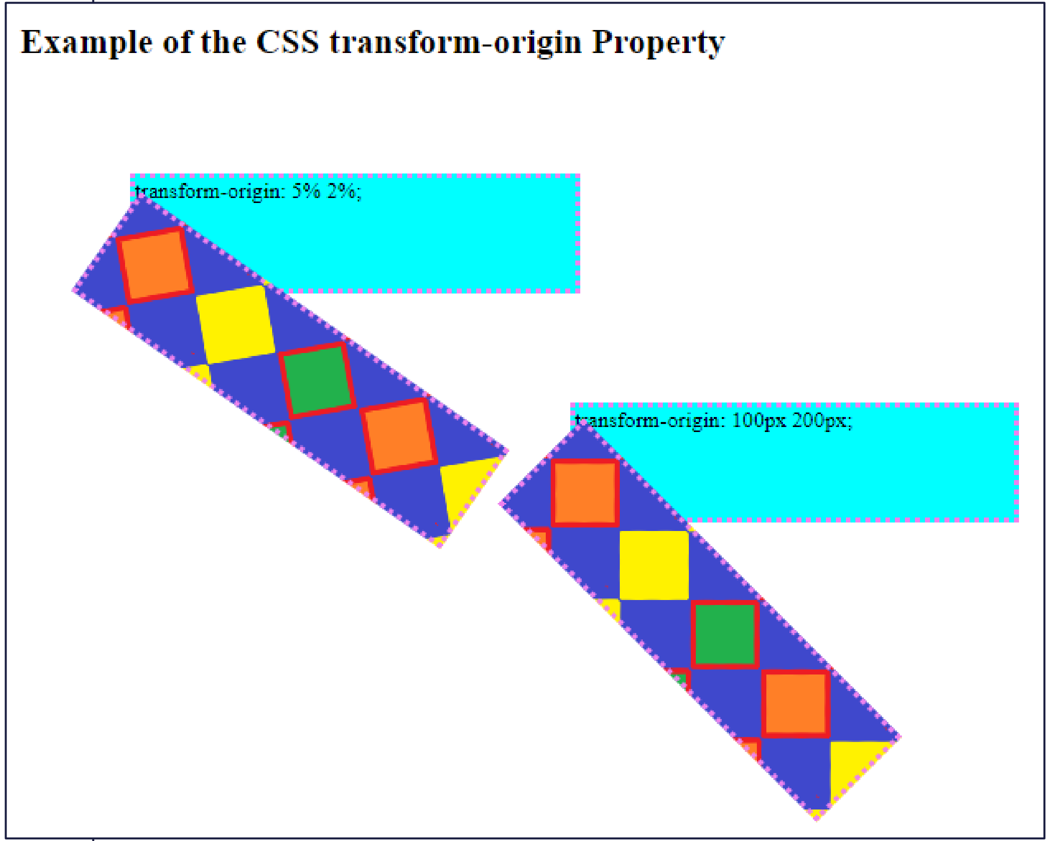
CSS transform-origin cont’d

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

Values	Description
x-axis	It represents the horizontal position. This value specifies the position of the view at x-axis. Its possible values are length, percentage, **left, right**, and **center**.
y-axis	It represents the vertical position. This value specifies the position of the view at y-axis. Its possible values are **length, percentage, top, bottom,** and **center**.
z-axis	This value is used with the 3D transformations. This value specifies the position of the view at z-axis. It can be defined using the **length** values. It does not allow the percentage values.
initial	It sets the property to its default value.
inherit	It inherits the property from its parent element.

CSS transform-origin example

```
…
<style>
div{
height: 100px;
width: 400px;
border: 5px dotted violet;
font-size: 20px;
}
.outer {  margin: 100px;
background-color: cyan;   }
.box {
background: url( "diamond.png");
transform: rotate(35deg);
transform-origin: 5% 2%;
}
.outer1{  margin-left: 500px;
background-color: cyan;  }
.box1 { background: url( "diamond.png");
transform: rotate(45deg);
transform-origin: 5% 2%; }
…..
<body>
<h1> Example ..Property </h1>
<div class="outer"> transform-origin: 5% 2%;
<div class="box"></div>
</div>
<div class="outer1"> transform-origin: 100px 200px;
<div class="box1"></div>
</div>……
```



Example of the CSS transform-origin Property

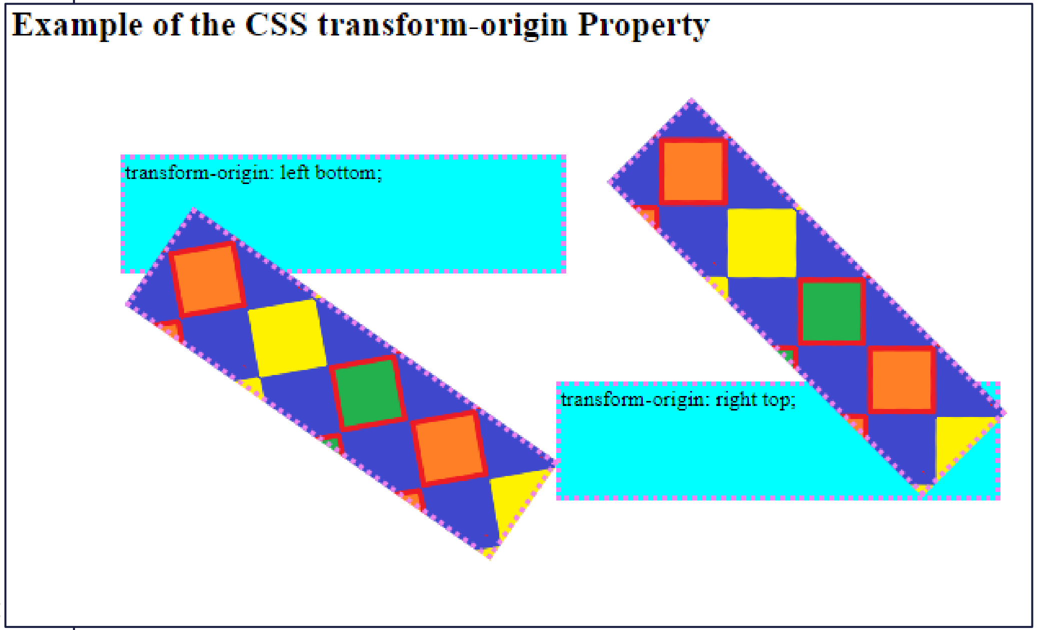
transform-origin: 5% 2%;

transform-origin: 100px 200px;

CSS transform-origin example

```
…
div{ height: 100px;
width: 400px;
border: 5px dotted violet;
font-size: 20px; }
.outer { margin: 100px;
background-color: cyan; }
.box { background: url( "diamond.png");
transform: rotate(35deg);
transform-origin: left bottom; }
.outer1{ margin-left: 500px;
background-color: cyan; }
.box1 {  background: url( "diamond.png");
transform: rotate(45deg);
transform-origin: right top; }
</style>
….
<body>
<h1> Example .. Property </h1>
<div class="outer"> transform-origin: left bottom;
<div class="box"></div>
</div>
<div class="outer1"> transform-origin: right top;
<div class="box1">….
```

transform-origin example using keyword values


Example of the CSS transform-origin Property

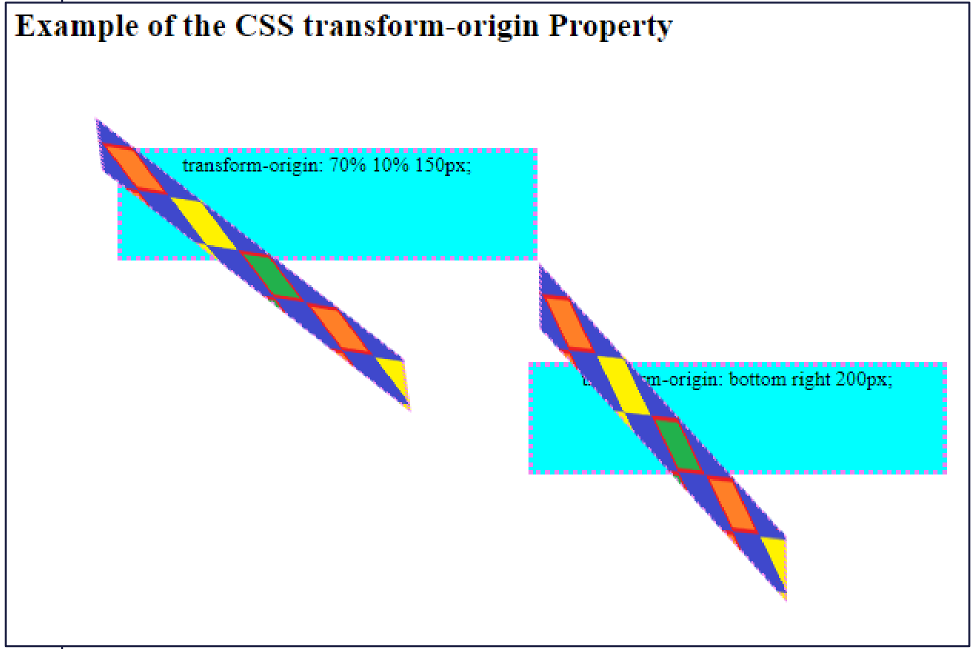
transform-origin: left bottom;

transform-origin: right top;

CSS transform-origin example

```
….
div{ height: 100px;
width: 400px;
border: 5px dotted violet;
font-size: 20px; }
.outer { margin: 100px;
background-color: cyan;
text-align: center; }
.box { background: url( "diamond.png");
  transform: rotate3d(3, 2, 1, 75deg);
  transform-origin: 70% 10% 150px; }
.outer1{ margin-left: 500px;
background-color: cyan;
text-align: center; }
.box1 { background: url( "diamond.png");
  transform: rotate3d(2, 2, 1, 75deg);
  transform-origin: bottom right 200px; }
……..
<body>
<h1> Example ….Property </h1>
<div class="outer"> transform-origin: 70% 10%
150px;
<div class="box"></div>
</div>
<div class="outer1"> transform-origin: bottom right
200px;
<div class="box1"></div>
</div>……..
```

transform-origin example property on the 3D transformed elements



Example of the CSS transform-origin Property

transform-origin: 70% 10% 150px;

transform-origin: bottom right 200px;

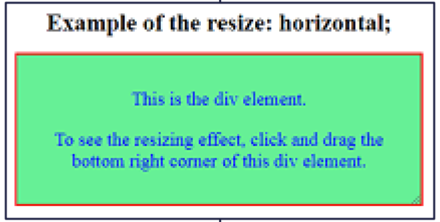
CSS resize property

- This CSS property allows the user to control the resizing of an element just by clicking or dragging the bottom right corner of the element
- This CSS property is used to define how an element is resizable by the user
- It doesn't apply on the block or inline elements where overflow is set to visible. So, to control the resizing of an element, we have to set the overflow other than visible like (overflow: hidden or overflow: scroll)
- Syntax
  - resize: none | horizontal | vertical | both | initial | inherit;
- **none**: It is the default value of this property, which does not allow resizing the element.
- **horizontal**: This value allows the user to resize the element's width. It resizes the element in a horizontal direction
  - There is a unidirectional horizontal mechanism for controlling the width of an element
- **vertical**: It allows the user to resize the height of an element. It resizes the element in a vertical direction.
  - There is a unidirectional vertical mechanism for controlling the height of an element
- **both**: It allows the user to resize the width and height of an element. It resizes the element in both horizontal and vertical directions

CSS resize example

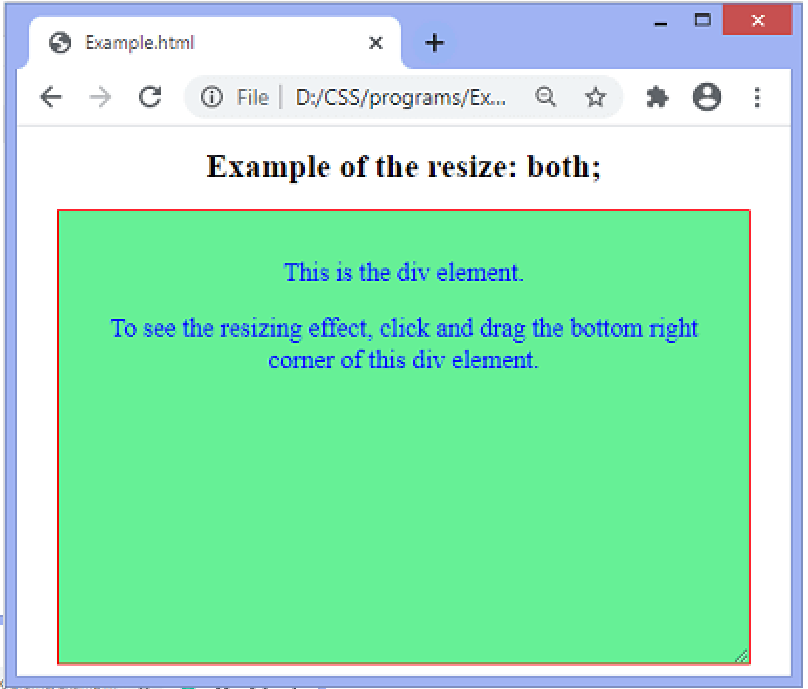
Example using horizontal value

```
.......
div {
  border: 2px solid red;
  padding: 20px;
  font-size: 25px;
  width: 300px;
  resize: horizontal;
  overflow: auto;
  background-color: lightgreen;
  color: blue;
}
</style>
......
<h1>Example of the resize: horizon
tal; </h1>
 <div>
  <p> This is the div element. </p>
  <p> To see the ...lement. </p>
</div>
...
```


Example of the resize: horizontal;

This is the div element.

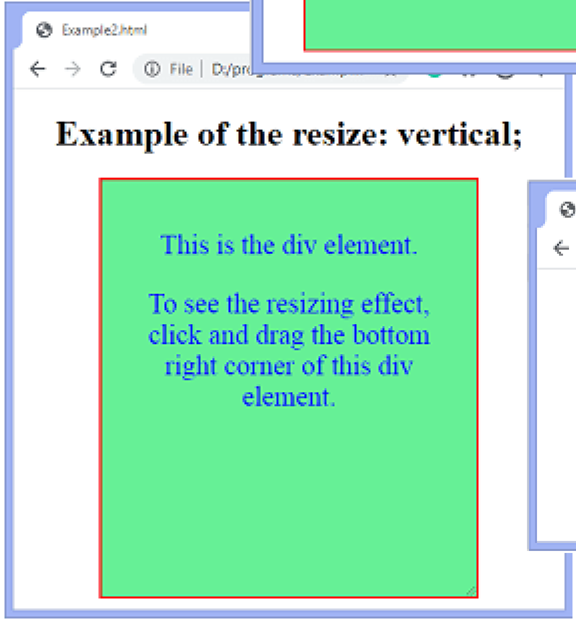
To see the resizing effect, click and drag the bottom right corner of this div element.



Example of the resize: both;

This is the div element.

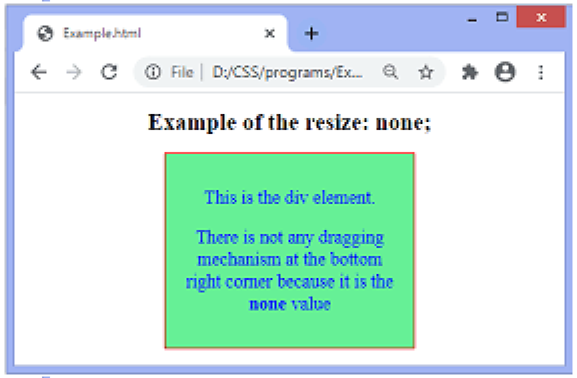
To see the resizing effect, click and drag the bottom right corner of this div element.



Example of the resize: vertical;

This is the div element.

To see the resizing effect, click and drag the bottom right corner of this div element.



Example of the resize: none;

This is the div element.

There is not any dragging mechanism at the bottom right corner because it is the none value

CSS transition-delay property

- This CSS property specifies the duration to start the transition effect
- The value of this property is defined as either seconds (s) or milliseconds (ms
- Syntax
  - transition-delay: time | initial | inherit;
- **time**: It specifies the amount of time (in seconds or milliseconds) to wait before the transition starts

CSS transition-delay example

```
…
<style>
div{
width: 100px;
height: 100px;
background: lightblue;
transition-property: background-color;
transition-duration: 1s;
transition-delay: 0.5s;

/* For Safari browser */
-webkit-transition-property: background-color;
-webkit-transition-duration: 1s;
-webkit-transition-delay: 0.5s;
}
div:hover {  background-color: brown;  }
</style>
……
<div></div>
<p> Move the cursor ..n effect. </p>  ….
```

- Example using the **transition-property, transition-duration,** and **transition-delay** properties
- There is a delay of **0.5s** to start the transition effect, i.e., the background color of the div element will be changed after the given amount of time.

CSS transition-delay example

```
….
<style>
div {
padding:15px;
width: 200px;
height: 200px;
background: lightgreen;
transition: background-
color 1s, width 2s, height 2s, transform 2s;
transition-delay: 1.5ms;
}
p{  font-size: 20px;  }
div:hover {
width: 300px;
height: 300px;
-webkit-transform:  rotate(360deg);
transform: rotate(360deg);
background-color: orange;
}
</style>
…..
<div>
<h2>JavaTpoint.com</h2>
<p> (Move your cursor on me to see the effect)</p></div
> ..
```

Example

- Transition effects on width, height, and transformation
- There is also a **transition-delay** in the **milliseconds (ms),** i.e., **2.5ms**.

Exercise

- Checkbox styling
- Radio button styling
- Navigation bar
- CSS overlay

?