CSS (Cascading Style Sheets) Notes

CSS stands for Cascading Style Sheets. It is the language for describing the presentation of Web pages, including colours, layout, and fonts, thus making our web pages presentable to the users.

CSS is designed to make style sheets for the web. It is independent of HTML and can be used with any XML-based markup language. Now let's try to break the acronym:

- Cascading: Falling of Styles
- Style: Adding designs/Styling our HTML tags
- Sheets: Writing our style in different documents

CSS Syntax:

```
Selector {
                Property 1 : value;
                        Property 2 : value;
                Property 3 : value;
            }
 For example:
        h1
           {
               Color: red;
                Text-align: center;

           }
         #unique
          {
                color: green;
          }
```

- Selector: selects the element you want to target
- Always remains the same whether we apply internal or external styling
- There are few basic selectors like tags, id's, and classes
- All forms this key-value pair
- Keys: properties(attributes) like color, font-size, background, width, height,etc

- Value: values associated with these properties

Types Off CSS:

There are 3 ways to write CSS in our HTML file.

- Inline CSS
- Internal CSS
- External CSS

Priority order : Inline > Internal > External

1. Inline CSS :

The style attribute works in the same way as any other HTML attribute. We use style, followed by the equality sign (=), and then a quote where all of the style values will be stored using the standard CSS property-value pairs - "property: value;"

<h1 style="...">...</h1>

Example : <h1 style="color: red; font-size: 40px;">Hello World</h1>

2. Internal CSS

Internal CSS (or Embedded CSS) refers to the method of adding CSS styles directly within the HTML document. It is Placed within the <style> tag in the <head> section of an HTML document.

```
<!DOCTYPE html>
<html>
<head>
  <style>
    /* Internal CSS styles go here */
  </style>
</head>
<body>
  <!-- HTML content goes here -->
</body>
</html>
```

3. External CSS

External CSS involves placing styles in a separate CSS file and linking it to an HTML document. The CSS file has a .css extension and is linked in the <head> section of the HTML file.

```
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" type="text/css" href="styles.css">
</head>
<body>
  <!-- HTML content goes here -->
</body>
</html>
```

Selectors in CSS:

In CSS (Cascading Style Sheets), selectors are patterns used to target and style specific HTML elements within a document. Selectors define the criteria for selecting one or more elements to which a particular style or set of styles will be applied.

There are 3 types of selectors:

1. Element Selector

2. ID Selector

3. Class Selector

Priority of Selectors : Id > Class>Element

1. Element Selector :

The element selector targets HTML elements based on their element type or tag name. When you want to apply a style to all instances of a specific HTML element, such as paragraphs (<p>), headings (<h1>, <h2>, etc.), or links (<a>).

element {/* styles */} --> Syntax

2. Class Selector('.') :

The class selector targets HTML elements based on their class attribute.When you want to apply a style to multiple elements that share the same class. Useful for styling groups of elements in a consistent way.

.classname {/* styles */} --> Syntax

3. ID Selector('#') :

The ID selector targets a specific HTML element based on its id attribute.When you want to uniquely style a particular element. Note: IDs should be unique within a page, so this selector is typically used for single, distinct elements.

#idname {/* styles */} --> Syntax

CSS TYPOGRAPHY :

CSS typography refers to the application of styles and formatting to text content on a web page using Cascading Style Sheets (CSS). It involves controlling the appearance, layout, and design of textual elements.

Properties Of CSS Typography :

1. CSS Colors :

```
  <!DOCTYPE html>
  <html>
  <head>
        <title>HTML</title>
        <link rel="stylesheet" type="text/css" href="first.css">
   <style>
        #center{   color:#ff0099;}
        h1{ color:rgba(255,0,0,0.5);}
    </style>
</head>
  <body>
          <h1>This heading will be green</h1>
   <p style="color:rgb(255,0,0)">This paragraph will be red</p>
          <p  id="center">This paragraph will be pink and center-
aligned</p>

   </body>
    </html>
```

2. CSS Background :

```
<!DOCTYPE html>
<html>
<head>
        <title>HTML</title>
        <link rel="stylesheet" type="text/css" href="first.css">
<style>

html{
         background: #ff9900;
}

p{
      background: url("https://encrypted-tbn0.gstatic.com/images?
q=tbn%3AANd9GcRT8t-o6oUJ-
E9YRhimOvTU2TSH7vlBnRWBN554_rX30dZah466&usqp=CAU");

    background-position:center;
        background-repeat:no-repeat;

        width: 100%;
        height: 600px;
}
</style>
</head>
<body>
<p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim
veniam,
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse
cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat
non
provident, sunt in culpa qui officia deserunt mollit anim id est laborum.
</p>
</body>
</html>
```

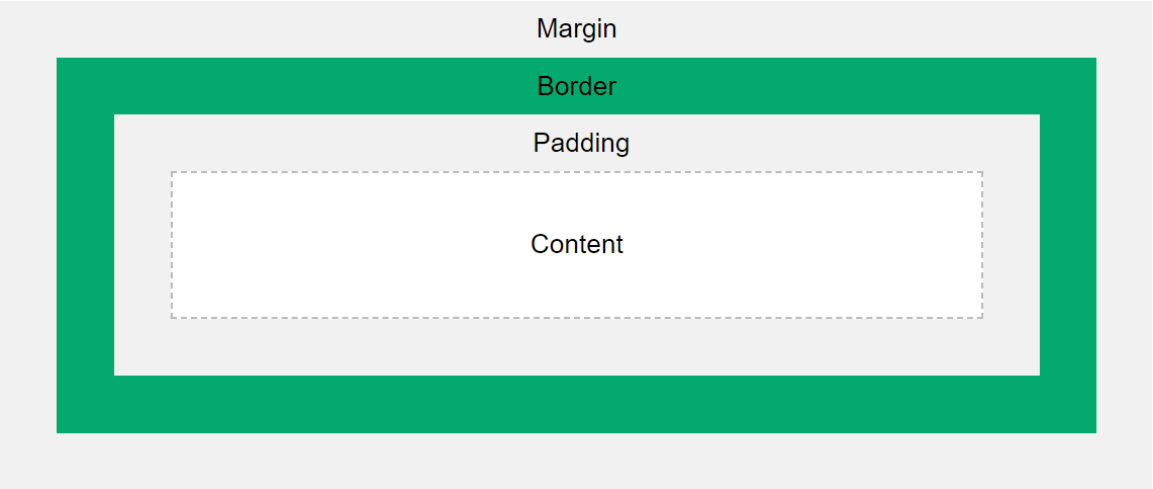
3. CSS Border :

```
<style>

p{
        border-style: solid;
        border-color: blue;
        border-width: 2px 5px;
        border-radius: 10%;
}



</style>
</head>
<body>
```

CSS BoxModel :

In CSS, the term "box model" is used when talking about design and layout.

The CSS box model is essentially a box that wraps around every HTML element. It consists of: content, padding, borders and margins.



Content : The content area consists of content like image, text, or other forms of media content. The height and width properties help to modify the box dimensions.

Padding: The padding area is the space around the content area and within the border-box. It can be applied to all sides of the box or to the specific, selected side(s) - top, right, bottom, and/or left.

Border: The border area surrounds the padding and the content, and can be applied to all the sides of the box or to selected side(s) - top, right, bottom, and/or left.

Margin : The margin area consists of space between the border and the margin. The margin does not possess its own background color and is completely transparent. It shows the background color of the element, like the body element.

Demonstration of the box model:

```
div {
  width: 300px;
  border: 15px solid green;
  padding: 50px;
  margin: 20px;
}
```

For setting the width & height properties of an element(for properly rendering the content in the browser), we need to understand the working of the CSS Box model.

While setting the width and height properties of an element with CSS, we have only set the width and height of the content area. We need to add padding, borders, and margins in order to calculate the full size of an element.

Display Property :

The display property in CSS is used to control the layout behavior of an HTML element. It determines how an element is rendered in the document flow, affecting its visibility, box type, and interaction with other elements. The display property can take various values, each influencing the element's presentation in different ways.

There Are 2 Type of display properties : 1. Inline Element 2. Block Element

1. Inline Element :

Inline elements are HTML elements that are formatted on the same line as the text and elements that come before and after them.

Common inline elements include:

-
- <a> (anchor)
- (bold)
- (italic)
- <img> (image)
-
 (line break)

By default, most HTML elements are either block or inline based on their type. The display property can be used to change the default display behavior.

```
span {
  display: block; /* Changes span to block-level element */
}
```

Inline elements do not respect the width and height properties. They only take up as much width as necessary for their content.

```
a {
  width: 100px; /* Does not affect the width of the link */
}
```

Inline elements do not respect top and bottom margin or padding. Horizontal margin and padding, however, affect the surrounding content.

```
span {
  margin-top: 10px; /* Does not create space above and below the span */
  padding-left: 20px; /* Creates space to the left of the span */
}
```

2. Block-level Elements :

Block-level elements in CSS are those that generate a block-level box in the document structure. These elements typically start on a new line and take up the full available width, extending from the left to the right edges of their containing element

Block-level elements are HTML elements that create a block box in the document flow. They usually start on a new line and stretch horizontally to fill the full width of their containing element.

Common block-level elements include:

-
- <p> (paragraph)
- <h1> to <h6> (headings)
- , <ol> (unordered and ordered lists)
- (list items)
- <table>
- <form>
- <header>, <footer>, <section>, <article> (HTML5 structural elements)

Block-level behavior is the default for many HTML elements, but the display property can be used to change the default behavior of an element.

```
span {
  display: block; /* Changes span to a block-level element */
}
```

Block-level elements, by default, take up the full width of their containing element. Their width can be controlled using the width property.

```
div {
  width: 50%; /* Sets the width of the div to 50% of its containing element
}
```

Block-level elements respect both vertical and horizontal margin and padding properties.

```
p {
  margin-top: 10px; /* Creates space above the paragraph */
  padding-left: 20px; /* Adds padding to the left of the paragraph */
}
```

3. Inline-Block Element :

An inline-block element in CSS is a hybrid between inline and block-level elements. It combines some features of both, allowing the element to behave like an inline element with respect to surrounding content while still retaining some block-level properties.

An inline-block element generates a block-level box in the document flow but retains some characteristics of an inline-level element. It does not start on a new line and only takes up as much width as necessary.

The display: inline-block; property is used to set an element to behave as an inline-block.

```
div {
  display: inline-block;
}
```

Similar to inline elements, an inline-block element only takes up as much width as necessary. However, unlike pure inline elements, you can set both width and height.

```
span {
  display: inline-block;
  width: 100px;
  height: 50px;
}
```

Like block-level elements, an inline-block respects both vertical and horizontal margin and padding.

```
img {
  display: inline-block;
  margin-right: 10px;
  padding: 5px;
}
```

CSS position Property :

The CSS position property is used to set position for an element. it is also used to place an element behind another and also useful for scripted animation effect.

You can position an element using the top, bottom, left and right properties. These properties can be used only after position property is set first. A position element's computed position property is relative, absolute, fixed or sticky.

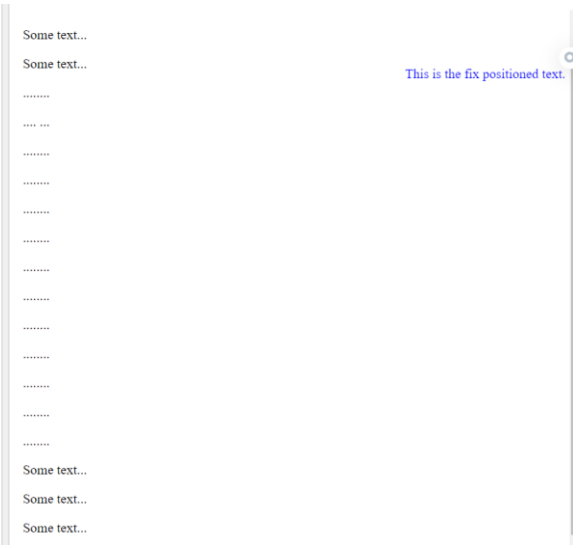
1) CSS Static Positioning :

This is a by default position for HTML elements. It always positions an element according to the normal flow of the page. It is not affected by the top, bottom, left and right properties.

2) CSS Fixed Positioning :

The fixed positioning property helps to put the text fixed on the browser. This fixed test is positioned relative to the browser window, and doesn't move even you scroll the window.

```
<!DOCTYPE html>
<html>
<head>
<style>
p.pos_fixed {
    position: fixed;
    top: 50px;
    right: 5px;
    color: blue;
}
</style>
</head>
<body>

<p>Some text...</p><p>Some text...</p><p>Some text...</p><p>........</p><p>....
...</p
><p>........</p><p>........</p><p>........</p><p>........</p>
<p>........ </p><p>........</p><p>........</p><p>........</p><p>........</p>
<p>........</p><p>........</p><p>Some text...</p><p>Some text...</p><p>Some
text...</p>
<p class="pos_fixed">This is the fix positioned text.</p>
</body>
</html>
```

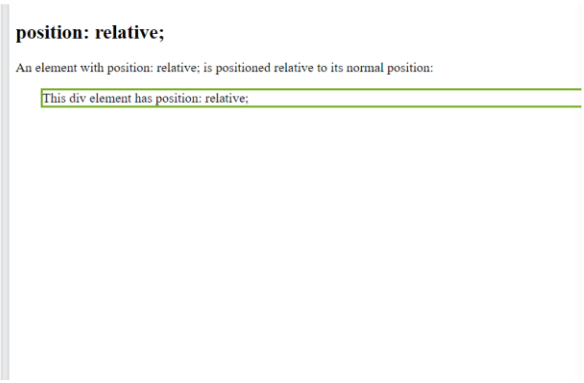


3) CSS Relative Positioning :

The relative positioning property is used to set the element relative to its normal position.

Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position. Other content will not be adjusted to fit into any gap left by the element.

```
<!DOCTYPE html>
<html>
<head>
<style>
div.relative {
  position: relative;
  left: 30px;
  border: 3px solid #73AD21;
}
</style>
</head>
<body>

<h2>position: relative;</h2>

<p>An element with position: relative; is positioned relative to its normal
position:</p>

<div class="relative">
This div element has position: relative;
</div>

</body>
</html>
```



4) CSS Absolute Positioning :

The absolute positioning is used to position an element relative to the first parent element that has a position other than static. If no such element is found, the containing block is HTML.

With the absolute positioning, you can place an element anywhere on a page.

```
<!DOCTYPE html>
<html>
<head>
<style>
h2 {
   position: absolute;
   left: 150px;
   top: 250px;
}
</style>
</head>
<body>
<h2>This heading has an absolute position</h2>
<p> The heading below is placed 150px from the left and 250px from the top of the page.</p>
</body>
</html>
```

5) CSS Sticky Positioning :

An element with position: sticky; is positioned based on the user's scroll position.

A sticky element toggles between relative and fixed, depending on the scroll position. It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like position:fixed).

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

CSS Units

There are various units in CSS to express the measurement and length. A CSS unit is used to determine the property size, which we set for an element or its content. The units in CSS are required to define the measurement such as margin: 20px; in which the px (or pixel) is the CSS unit. They are used to set margin, padding, lengths, and so on.

We cannot apply the whitespace between the number and the unit. The unit can be omitted for the value 0. Some properties of CSS allow the negative values of length.

The length unit in CSS is of two types: 1) Absolute length 2) Relative length.

1) Absolute length :

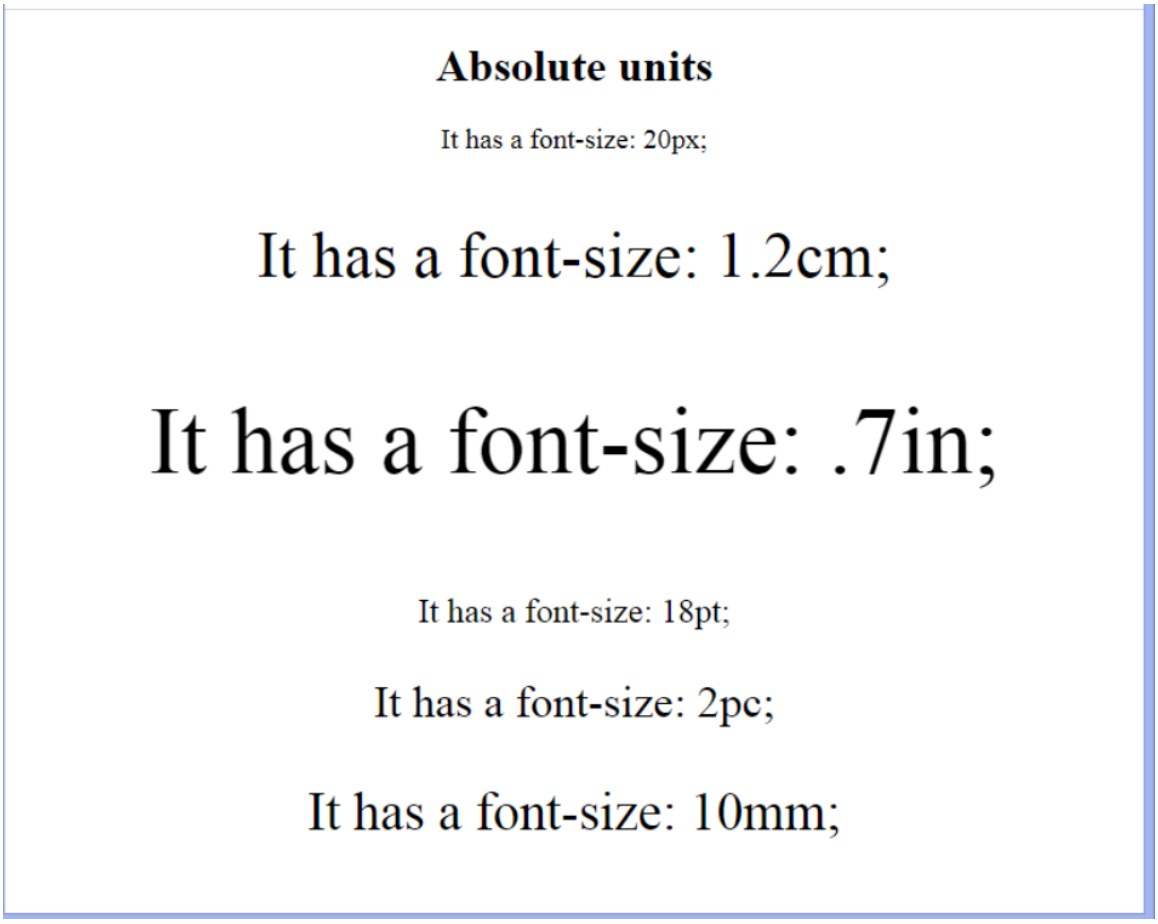
These are the fixed-length units, and the length expressed using the absolute units will appear as exactly that size. It is not recommended to use on-screen, because the size of the screen varies too much. So, the absolute units should be used when the medium of output is known, such as the print layout.

Absolute units are useful when the responsiveness is not considered in a project. They are less favorable for the responsive sites because they do not scale when the screen changes.

Generally, absolute lengths are considered to be the same size always.

Unit	Name	Explanation
cm	Centimeters	It is used to define the measurement in centimeters.
mm	Millimeters	It is used to define the measurement in millimeters.
in	Inches	It is used to define the measurement in inches. 1in = 96px = 2.54cm
pt	Points	It is used to define the measurement in points. 1pt = 1/72 of 1 inch.
pc	Picas	It is used to define the measurement in picas. 1pc = 12pt so, there 6 picas is equivalent to 1 inch.
px	Pixels	It is used to define the measurement in pixels. 1px = 1/96th of inch

```
<!DOCTYPE html>
<html>
<head>
<style>
body{
text-align: center;
}
</style>
</head>
<body>
<h1> Absolute units </h1>
<p style = "font-size: 20px;" > It has a font-size: 20px; </p>
<p style = "font-size: 1.2cm;" > It has a font-size: 1.2cm; </p>
<p style = "font-size: .7in;" > It has a font-size: .7in; </p>
<p style = "font-size: 18pt;" > It has a font-size: 18pt; </p>
<p style = "font-size: 2pc;" > It has a font-size: 2pc; </p>
<p style = "font-size: 10mm;" > It has a font-size: 10mm; </p>
</body>
</html>
```



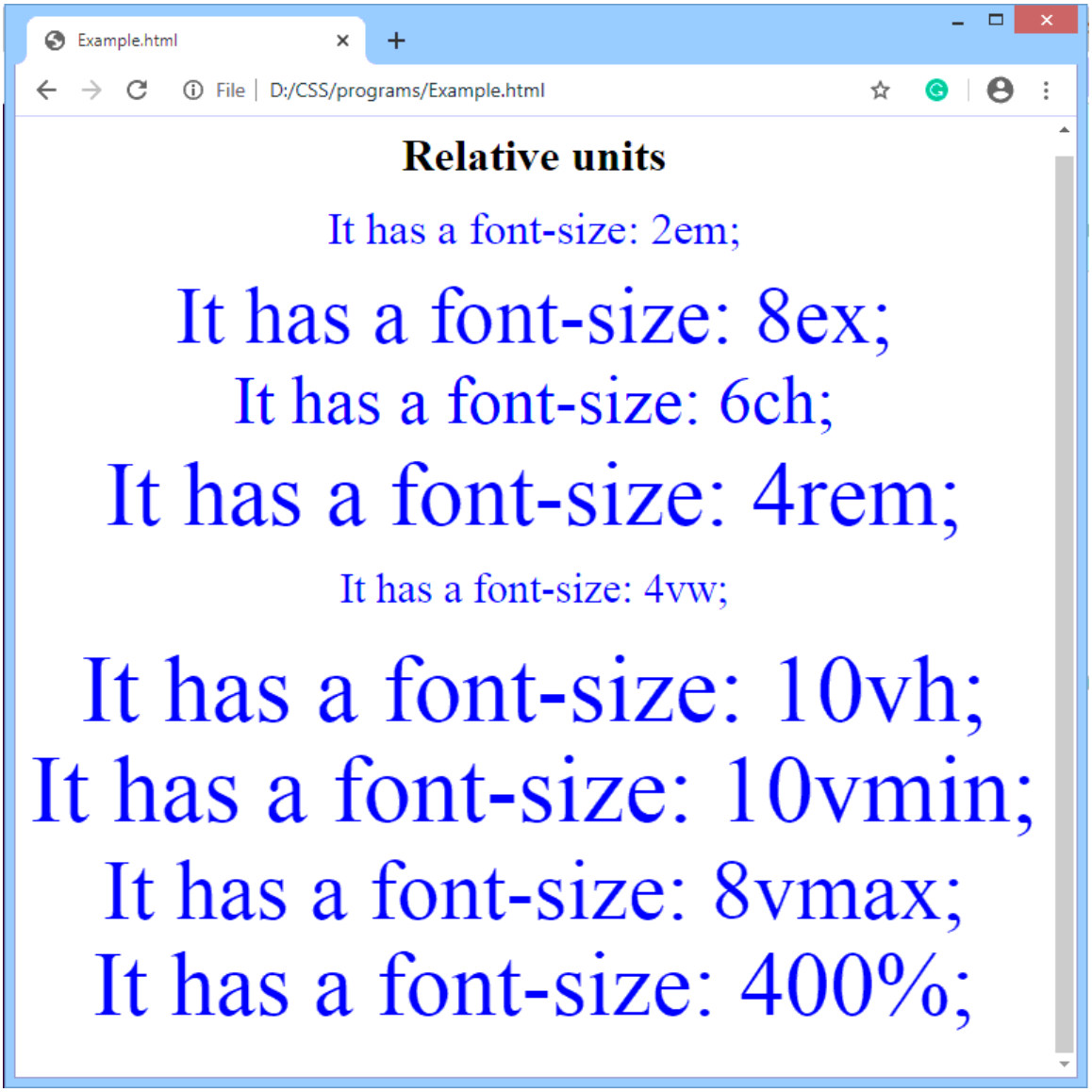
2) Relative length :

Relative units are good to style the responsive site because they scale relative to the window size or the parent. They specify the length, which is relative to another length property.

Depending on the device, if the size of the screen varies too much, then the relative length units are the best because they scale better between the different rendering mediums. We can use the relative units as the default for the responsive units. It helps us to avoid update styles for different screen sizes.

Unit	Name
em	It is relative to the font-size of the element.
ex	It is relative to the x-height of the font of the element. It is rarely used. The x-height is determined by the height of the lowercase letter 'x'.
ch	It is similar to the unit ex, but instead of using the height of the letter x, it measures the width of the integer "0" (zero).
rem	It is the font-size of the root element
vh	It is relative to the height of the viewport. 1vh = 1% or 1/100 of the height of the viewport.
vw	It is relative to the width of the viewport. 1vw = 1% or 1/100 of the width of viewport
vmin	It is relative to the smaller dimension of the viewport. 1vmin = 1% or 1/100 of the viewport's smaller dimension.
vmax	It is relative to the larger dimension of the viewport. 1vmax = 1% or 1/100 of the viewport's larger dimension.
%	It is used to define the measurement as a percentage that is relative to another value.

```
<!DOCTYPE html>
<html>
<head>
<style>
body{
text-align: center;
}
p{
line-height: 0.1cm;
color: blue;
}
</style>
</head>
<body>
<h1> Relative units </h1>
<p style = "font-size: 2em;" > It has a font-size: 2em; </p>
<p style = "font-size: 8ex;" > It has a font-size: 8ex; </p>
<p style = "font-size: 6ch;" > It has a font-size: 6ch; </p>
<p style = "font-size: 4rem;" > It has a font-size: 4rem; </p>
<p style = "font-size: 4vw;" > It has a font-size: 4vw; </p>
<p style = "font-size: 10vh;" > It has a font-size: 10vh; </p>
<p style = "font-size: 10vmin;" > It has a font-size: 10vmin; </p>
<p style = "font-size: 8vmax;" > It has a font-size: 8vmax; </p>
<p style = "font-size: 400%;" > It has a font-size: 400%; </p>
</body>
</html>
```


Example.html
File | D:/CSS/programs/Example.html
Relative units
It has a font-size: 2em;
It has a font-size: 8ex;
It has a font-size: 6ch;
It has a font-size: 4rem;
It has a font-size: 4vw;
It has a font-size: 10vh;
It has a font-size: 10vmin;
It has a font-size: 8vmax;
It has a font-size: 400%;