

CSS(Cascading Style Sheet)

- CSS stands for Cascading Style Sheets
 - CSS describes how HTML elements are to be displayed on screen, paper, or in other media
 - CSS saves a lot of work. It can control the layout of multiple web pages all at once
 - External stylesheets are stored in CSS files
- CSS Syntax

The selector points to the HTML element you want to style.

The declaration block contains one or more declarations separated by semicolons.

Each declaration includes a CSS property name and a value, separated by a colon.

Multiple CSS declarations are separated with semicolons, and declaration blocks are surrounded by curly braces.

CSS Selectors

CSS selectors are used to "find" (or select) the HTML elements you want to style.

We can divide CSS selectors into five categories:

- Simple selectors (select elements based on name, id, class)
- Combinator selectors (select elements based on a specific relationship between them)
- Pseudo-class selectors (select elements based on a certain state)
- Pseudo-elements selectors (select and style a part of an element)
- Attribute selectors (select elements based on an attribute or attribute value)

The CSS element Selector

The element selector selects HTML elements based on the element name.

The CSS id Selector

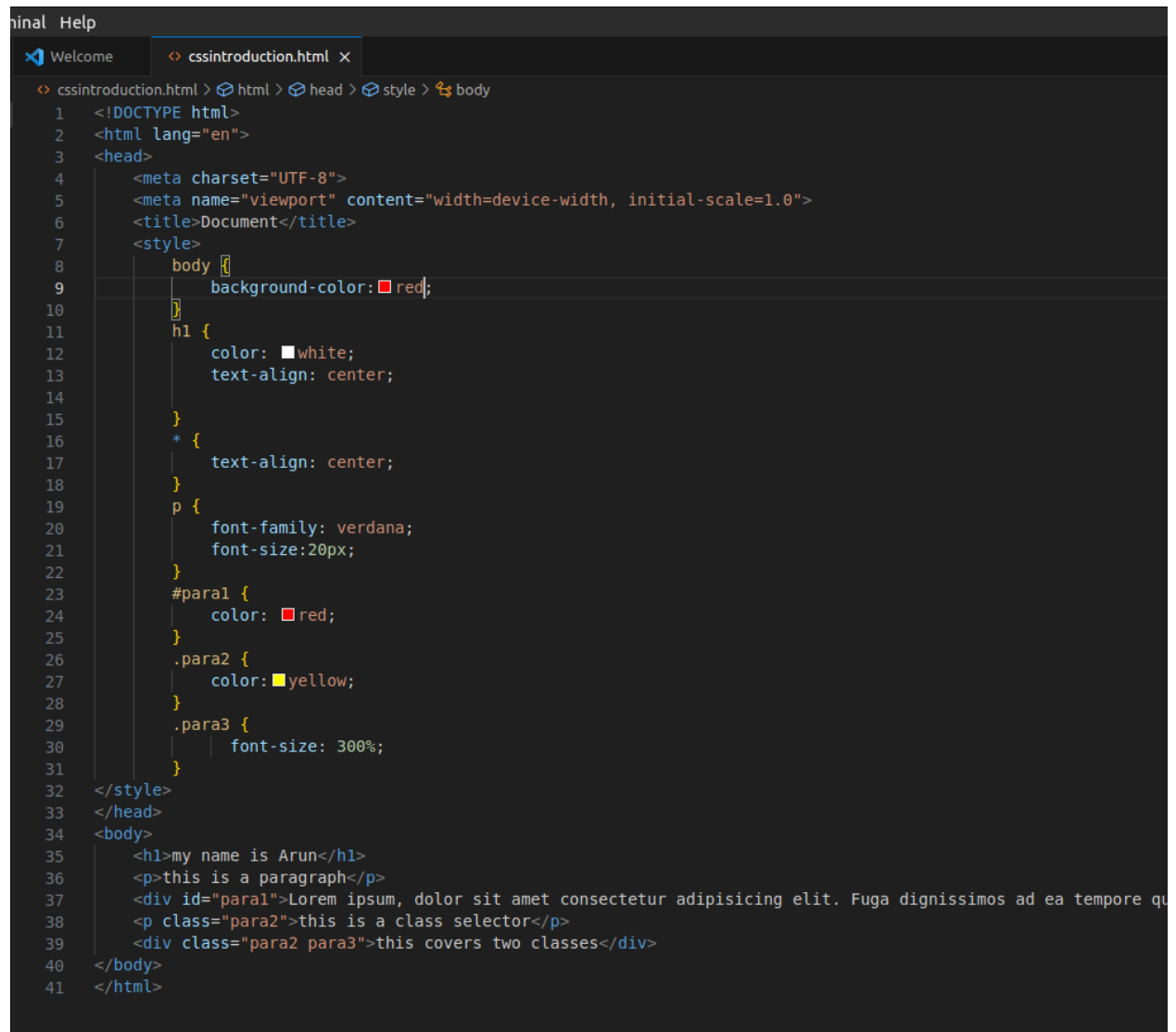
The id selector uses the id attribute of an HTML element to select a specific element.

The CSS class Selector

The class selector selects HTML elements with a specific class attribute. To select elements with a specific class, write a period (.) character, followed by the class name.

The CSS Universal Selector

The universal selector (*) selects all HTML elements on the page.



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7   <style>
8     body {
9       background-color: red;
10    }
11    h1 {
12      color: white;
13      text-align: center;
14    }
15    * {
16      text-align: center;
17    }
18    p {
19      font-family: verdana;
20      font-size: 20px;
21    }
22    #para1 {
23      color: red;
24    }
25    .para2 {
26      color: yellow;
27    }
28    .para3 {
29      font-size: 300%;
30    }
31  </style>
32 </head>
33 <body>
34   <h1>my name is Arun</h1>
35   <p>this is a paragraph</p>
36   <div id="para1">Lorem ipsum, dolor sit amet consectetur adipisicing elit. Fuga dignissimos ad ea tempore qu
37   <p class="para2">this is a class selector</p>
38   <div class="para2 para3">this covers two classes</div>
39 </body>
40 </html>
```

my name is Arun

this is a paragraph

this is a class selector

this covers two classes

Three Ways to Insert CSS

There are three ways of inserting a style sheet:

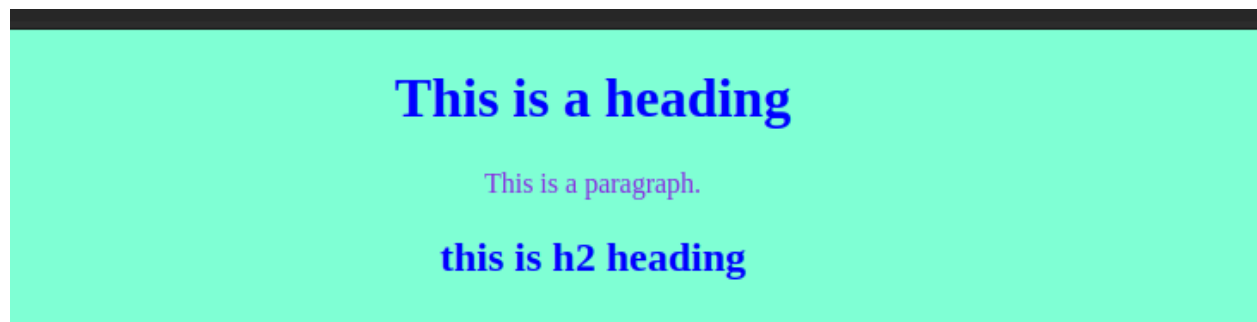
- External CSS
- Internal CSS
- Inline CSS

External CSS

With an external style sheet, you can change the look of an entire website by changing just one file!

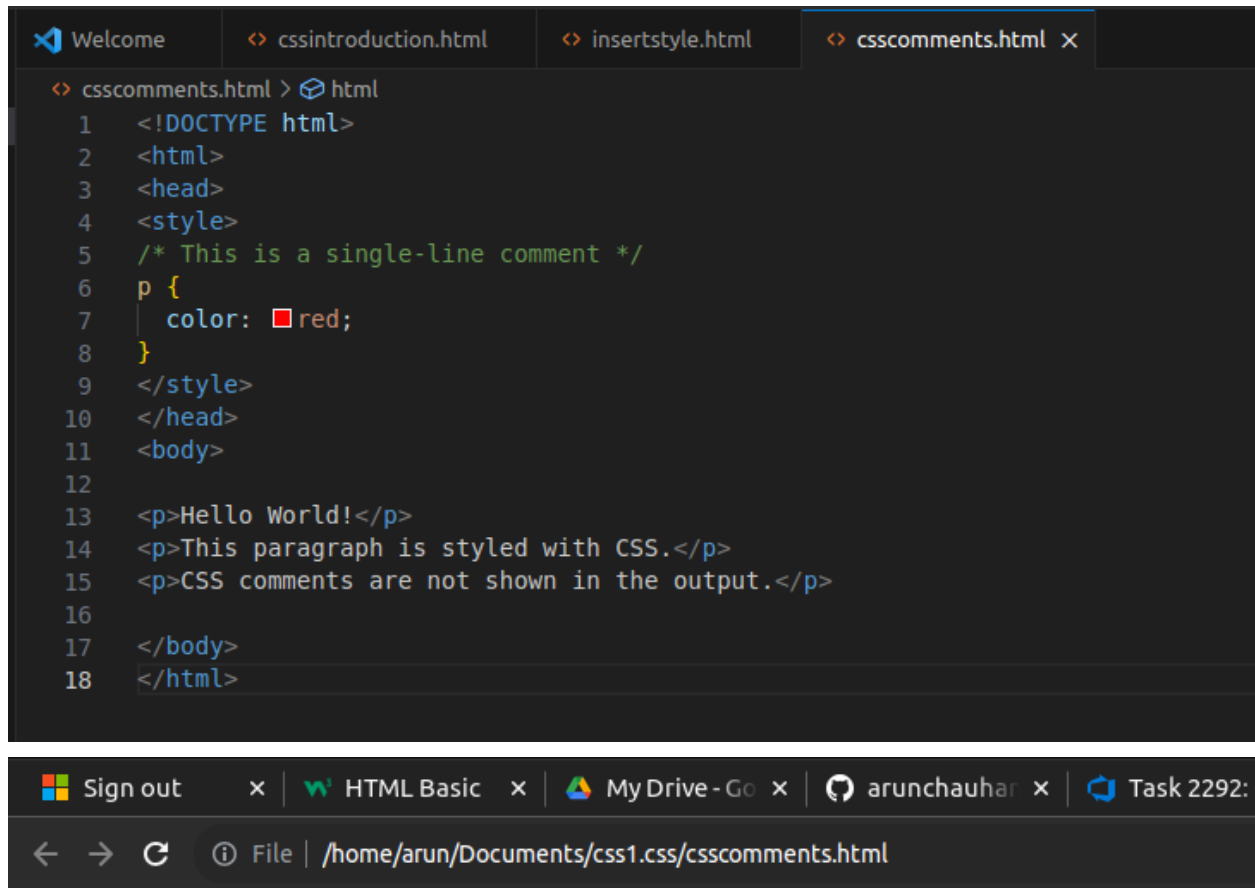
Each HTML page must include a reference to the external style sheet file inside the <link> element, inside the head section.

```
mat Help
Welcome  <> cssintroduction.html  <> insertstyle.html x
<> insertstyle.html > html > body > h1
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7      <link rel="style sheet" href="mystyle.css">
8      <style>
9          body{
10             background-color: aquamarine;
11          }
12          h1{
13             color: blue;
14             text-align: center;
15          }
16      </style>
17 </head>
18 <body>
19     <h1>This is a heading</h1>
20     <p style="color: blueviolet; text-align:center">This is a paragraph.</p>
21     <h2 style="color: blue;text-align:center;">this is h2 heading</h2>
22
23 </body>
24 </html>
```



CSS Comments

A CSS comment is placed inside the <style> element, and starts with /* and ends with */.



The screenshot shows a web browser window with a dark theme. The top bar contains several tabs: 'Welcome', 'cssintroduction.html', 'insertstyle.html', and 'csscomments.html' (which is the active tab). Below the tabs is a code editor showing the content of 'csscomments.html'. The code is as follows:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <style>
5 /* This is a single-line comment */
6 p {
7   color: red;
8 }
9 </style>
10 </head>
11 <body>
12
13 <p>Hello World!</p>
14 <p>This paragraph is styled with CSS.</p>
15 <p>CSS comments are not shown in the output.</p>
16
17 </body>
18 </html>
```

Below the code editor, the browser's address bar shows the file path: `/home/arun/Documents/css1.css/csscomments.html`. The rendered output of the HTML is displayed below the address bar, showing three paragraphs of text, all of which are colored red due to the CSS rule.

Hello World!

This paragraph is styled with CSS.

CSS comments are not shown in the output.

CSS Colors

Colors are specified using predefined color names, or RGB, HEX, HSL, RGBA, HSLA values.

CSS Background Color

CSS Text Color

CSS Color Values

In CSS, colors can also be specified using RGB values, HEX values, HSL values, RGBA values, and HSLA values:

Same as color name "Tomato":

`rgb(255, 99, 71)`

#ff6347

hsl(9, 100%, 64%)

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

CSS Backgrounds

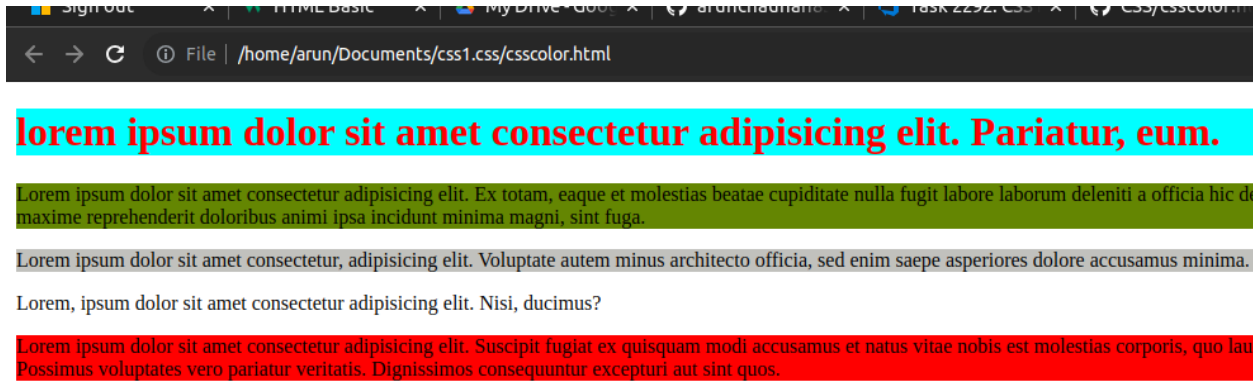
The CSS background properties are used to add background effects for elements.

CSS background-color

The background-color property specifies the background color of an element.

```

Welcome  < cssintroduction.html  < insertstyle.html  < csscomments.html  < csscolor.html x
csscolor.html > html > head > style > #para3
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7      <style>
8          #para1{
9              background-color: aqua;
10             color: red;
11         }
12         #para2{
13             background-color: rgb(100,134,2);
14         }
15         #para3{
16             background-color: rgba(132,132,123,0.5);
17         }
18         .hex{
19             background-color: #ffffff;
20         }
21         .hsl{
22             background-color: hsl(0,100%,50%);
23         }
24     </style>
25 </head>
26 <body>
27     <h1 id="para1">lorem ipsum dolor sit amet consectetur adipisicing elit. Pariatur, eum.</
28     <p id="para2">Lorem ipsum dolor sit amet consectetur adipisicing elit. Ex totam, eaque e
29     <p id="para3">Lorem ipsum dolor sit amet consectetur, adipisicing elit. Voluptate autem
30     <p class="hex">Lorem, ipsum dolor sit amet consectetur adipisicing elit. Nisi, ducimus?<
31     <p class="hsl">Lorem ipsum dolor sit amet consectetur adipisicing elit. Suscipit fugiat
32 </body>
33 </html>
```



CSS background-image

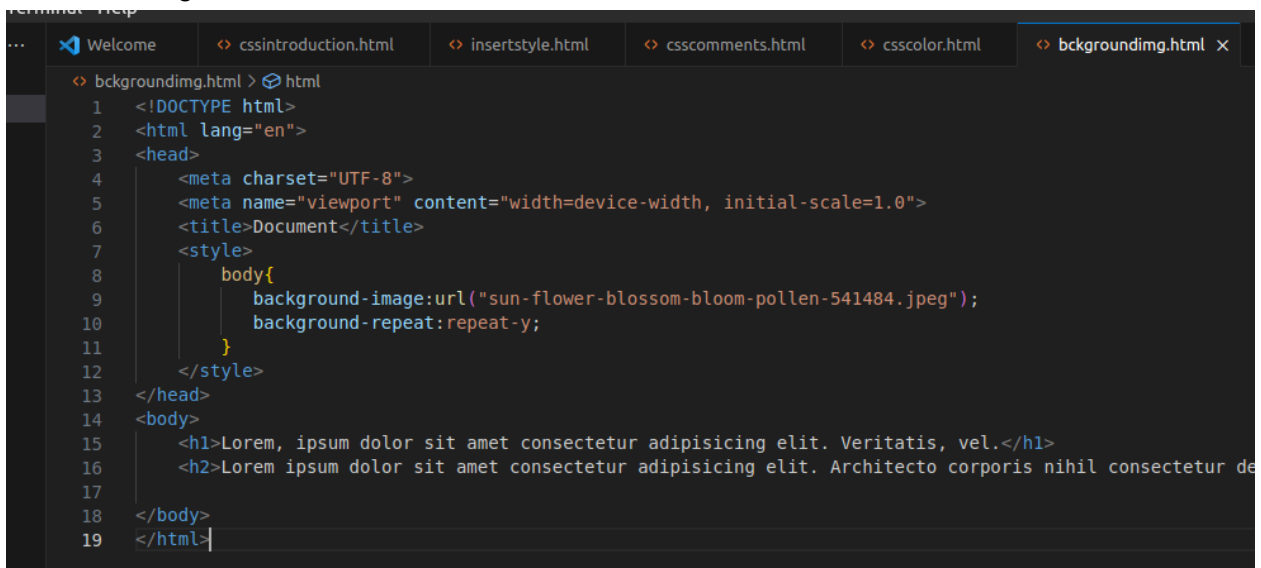
The background-image property specifies an image to use as the background of an element.

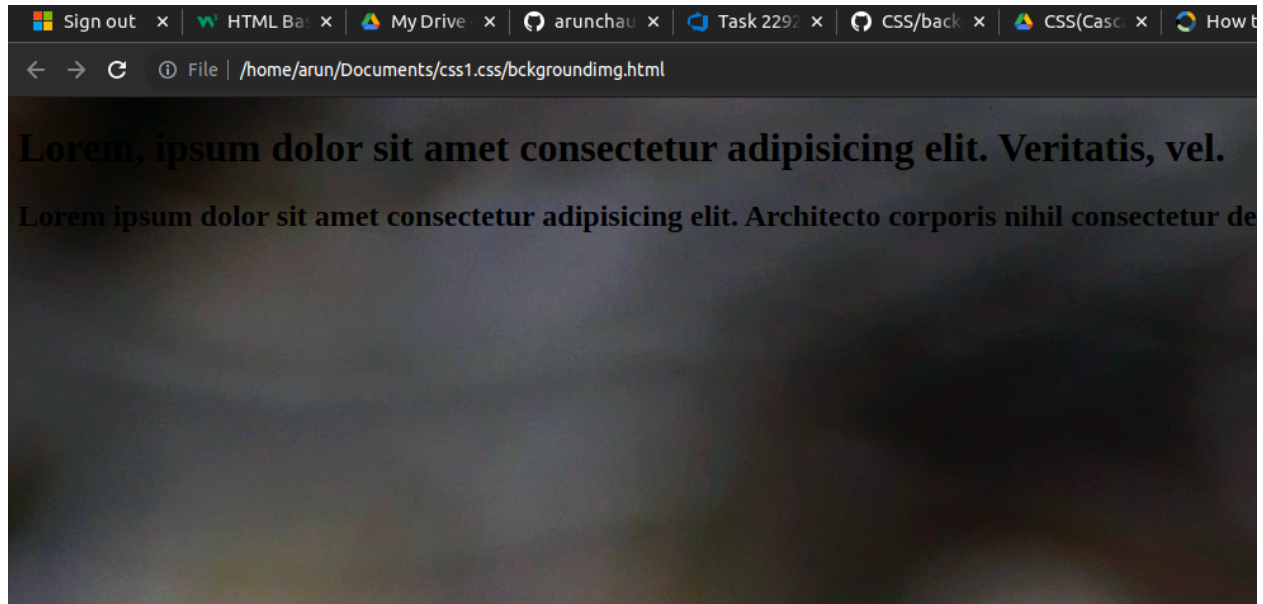
By default, the image is repeated so it covers the entire element.

CSS background-repeat

By default, the background-image property repeats an image both horizontally and vertically.

Some images should be repeated only horizontally or vertically, or they will look strange, like this:





CSS Borders

The CSS border properties allow you to specify the style, width, and color of an element's border.

CSS Border Style

The border-style property specifies what kind of border to display.

The following values are allowed:

- dotted - Defines a dotted border
- dashed - Defines a dashed border
- solid - Defines a solid border
- double - Defines a double border
- groove - Defines a 3D grooved border. The effect depends on the border-color value
- ridge - Defines a 3D ridged border. The effect depends on the border-color value
- inset - Defines a 3D inset border. The effect depends on the border-color value
- outset - Defines a 3D outset border. The effect depends on the border-color value
- none - Defines no border
- hidden - Defines a hidden border

CSS Border Width

The border-width property specifies the width of the four borders.

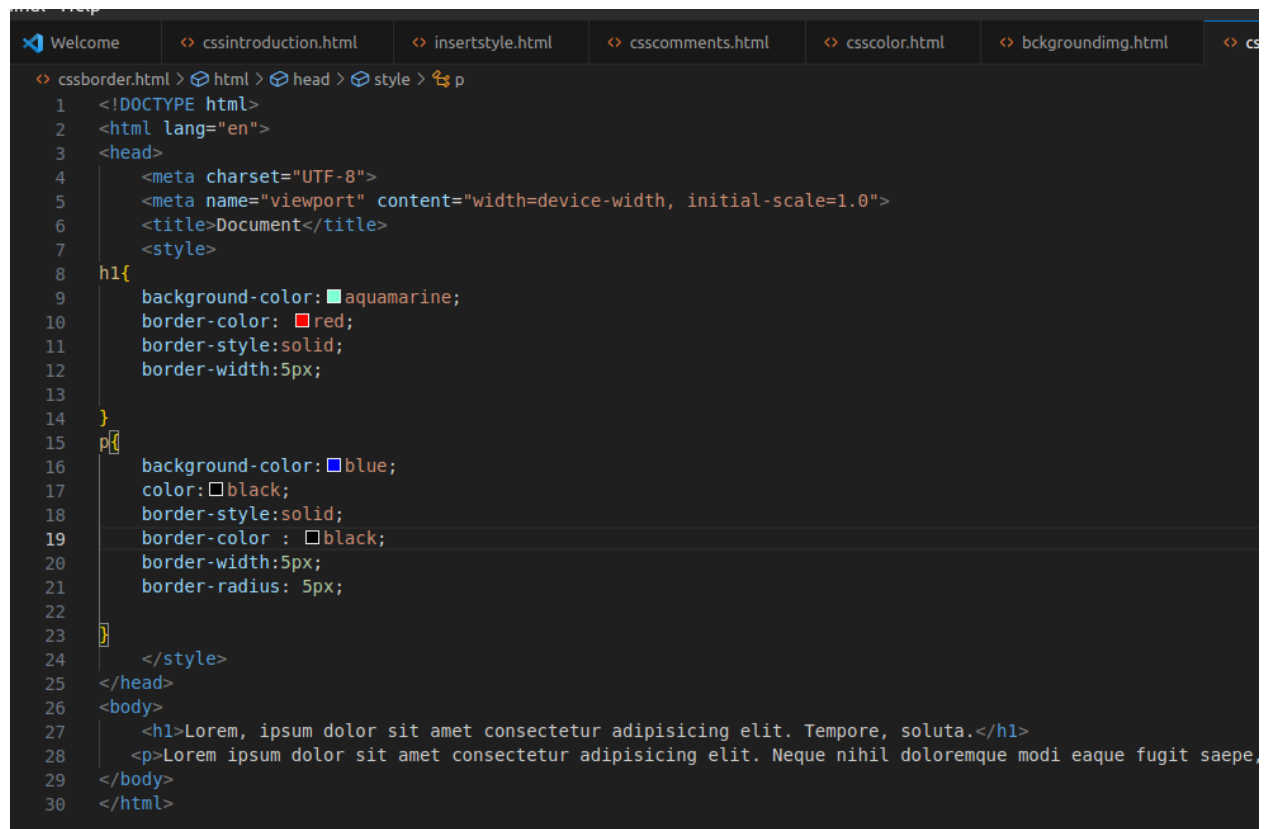
The width can be set as a specific size (in px, pt, cm, em, etc) or by using one of the three pre-defined values.

CSS Border Color

The border-color property is used to set the color of the four borders.

The color can be set by:

- name - specify a color name, like "red"
- HEX - specify a HEX value, like "#ff0000"
- RGB - specify a RGB value, like "rgb(255,0,0)"
- HSL - specify a HSL value, like "hsl(0, 100%, 50%)"
- transparent

A screenshot of a code editor with a dark theme. The editor has several tabs at the top: 'Welcome', 'cssintroduction.html', 'insertstyle.html', 'csscomments.html', 'csscolor.html', 'backgroundimg.html', and 'css'. The 'csscolor.html' tab is active. The code is written in a light blue font on a dark background. It shows a CSS file with a 'p' selector and an 'h1' selector. The 'p' selector has a blue background, black text, a solid black border, a 5px border width, and a 5px border radius. The 'h1' selector has an aquamarine background, a red border, a solid border style, and a 5px border width. The HTML file shows a document with a title 'Document', a head section with meta tags for charset and viewport, and a body section with an 'h1' and a 'p' element containing Lorem Ipsum text.

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7   <style>
8     h1{
9       background-color: aquamarine;
10      border-color: red;
11      border-style:solid;
12      border-width:5px;
13    }
14  }
15  p{
16    background-color: blue;
17    color: black;
18    border-style:solid;
19    border-color : black;
20    border-width:5px;
21    border-radius: 5px;
22  }
23 }
24 </style>
25 </head>
26 <body>
27   <h1>Lorem, ipsum dolor sit amet consectetur adipisicing elit. Tempore, soluta.</h1>
28   <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Neque nihil doloremque modi eaque fugit saepe,
29 </body>
30 </html>
```

Lorem, ipsum dolor sit amet consectetur adipisicing elit. Tempore, soluta.

Lorem ipsum dolor sit amet consectetur adipisicing elit. Neque nihil doloremque modi eaque fugit saepe, incidunt laudantium, provident laborum nulla, omnis molestiae, fugiat explicabo minus cupiditate ipsum, ducimus beatae quis inventore porro magnam, saepe facilis earum similique soluta veritatis. Unde totam blanditiis dolorem doloremque voluptatum, aliquid ipsum itaque ex sunt. A sed perspiciatis perferendis assumenda iste ratione, quibusdam repellendus debitis odit, error a

CSS Margins

The CSS margin properties are used to create space around elements, outside of any defined borders.

With CSS, you have full control over the margins. There are properties for setting the margin for each side of an element (top, right, bottom, and left).

Margin - Individual Sides

CSS has properties for specifying the margin for each side of an Element:

- margin-top
- margin-right
- margin-bottom
- margin-left

CSS Padding

The CSS padding properties are used to generate space around an element's content, inside of any defined borders.

With CSS, you have full control over the padding. There are properties for setting the padding for each side of an element (top, right, bottom, and left).

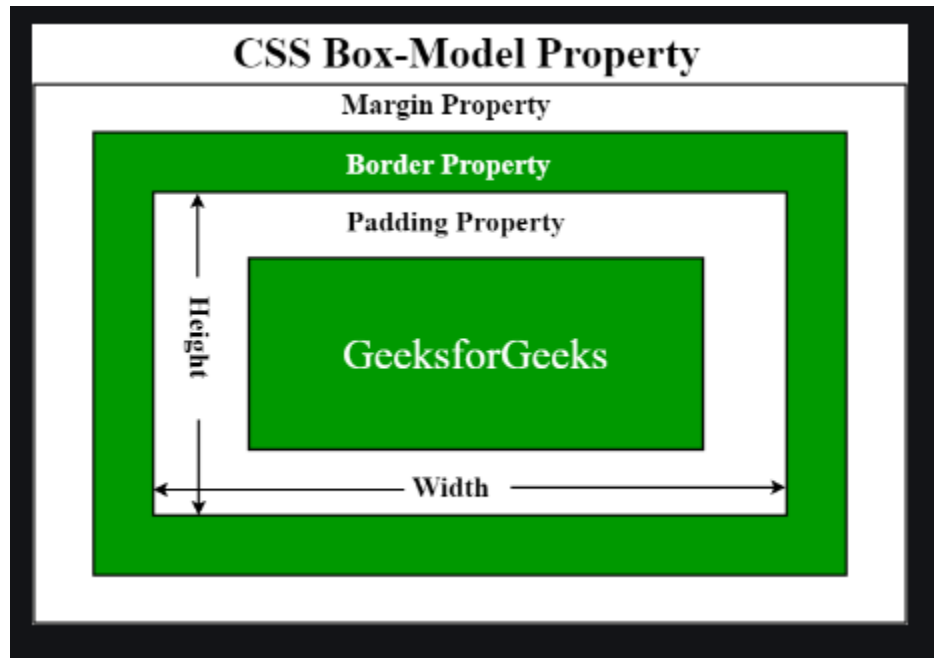
The CSS height and width properties are used to set the height and width of an element.

The CSS max-width property is used to set the maximum width of an Element.

The CSS Box Model

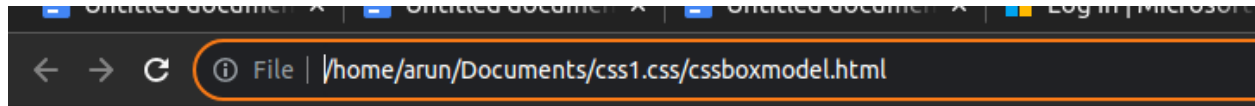
In CSS, the term "box model" is used when talking about design and Layout.

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



```
Welcome    <> cssintroduction.html    <> insertstyle.html    <> csscomments.html    <> csscolor.html

<> cssboxmodel.html > html > head > style > h1
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7      <style>
8          h1{
9              background-color: aquamarine;
10             border-color: red;
11             border-style:solid;
12             border-width:5px;
13             padding:20px;
14             margin:50px;
15             height:500px;
16             width:500px;
17         }
18     }
19     p{
20         background-color: blue;
21         color: black;
22         border-style:solid;
23         border-color : black;
24         border-width:5px;
25         border-radius: 5px;
26         padding-left:10px;
27         padding-right: 15px;
28         padding-top:15px;
29         padding-bottom:5px;
30         margin:10px;
31     }
32 }
33 </style>
34 </head>
35 <body>
36     <h1>Lorem, ipsum dolor sit amet consectetur adipisicing elit. Tempore, solu
37     <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Neque nihil dolo
38 </body>
39 </html>
```



**Lorem, ipsum dolor sit amet
consectetur adipisicing elit.
Tempore, soluta.**

Lorem ipsum dolor sit amet consectetur adipisicing elit. Neque nihil doloremque modi eaque fugit saepe, in voluptate molestiae, fugiat explicabo minus cupiditate ipsum, ducimus beatae quis inventore porro magnam beatae molestias dolorem doloremque voluptatum, aliquid ipsum itaque ex sunt. A sed perspiciatis perferend

Text Color

The color property is used to set the color of the text. The color is specified by:

- a color name - like "red"
- a HEX value - like "#ff0000"
- an RGB value - like "rgb(255,0,0)"

CSS Fonts

Font Family: Choosing the right font has a huge impact on how the readers experience a website. The right font can create a strong identity for your brand. Using a font that is easy to read is important. The font adds value to your text. It is also important to choose the correct color and text size for the font.

In CSS there are five generic font families:

- Serif fonts have a small stroke at the edges of each letter. They create a sense of formality and elegance.
- Sans-serif fonts have clean lines (no small strokes attached). They create a modern and minimalistic look.
- Monospace fonts - here all the letters have the same fixed width. They create a mechanical look.
- Cursive fonts imitate human handwriting.
- Fantasy fonts are decorative/playful fonts.

Font Web Safe: Web safe fonts are fonts that are universally installed across all browsers and devices. However, there are no 100% completely web safe fonts. There is always a chance that a font is not found or is not installed properly. Therefore, it is very important to always use fallback fonts. This means that you should add a list of similar "backup fonts" in the font-family property. If the first font does not work, the browser will try the next one, and the next one, and so on. Always end the list with a generic font family name.

Font Fallbacks: Below are some commonly used font fallbacks, organized by the 5 generic font families:

- Serif
- Sans-serif
- Monospace
- Cursive
- Fantasy

Font Style: The font-style property is mostly used to specify italic text. This property has three values:

- normal - The text is shown normally
- italic - The text is shown in italics
- oblique - The text is "leaning" (oblique is very similar to italic, but less supported)

Font Size: The font-size property sets the size of the text. Being able to manage the text size is important in web design. However, you should not use font size adjustments to make paragraphs look like headings, or headings look like paragraphs. Always use the proper HTML tags, like <h1> - <h6> for headings and <p> for paragraphs. The font-size value can be an absolute, or relative size.

Absolute size:

- Sets the text to a specified size
- Does not allow a user to change the text size in all browsers (bad for accessibility reasons)
- Absolute size is useful when the physical size of the output is known

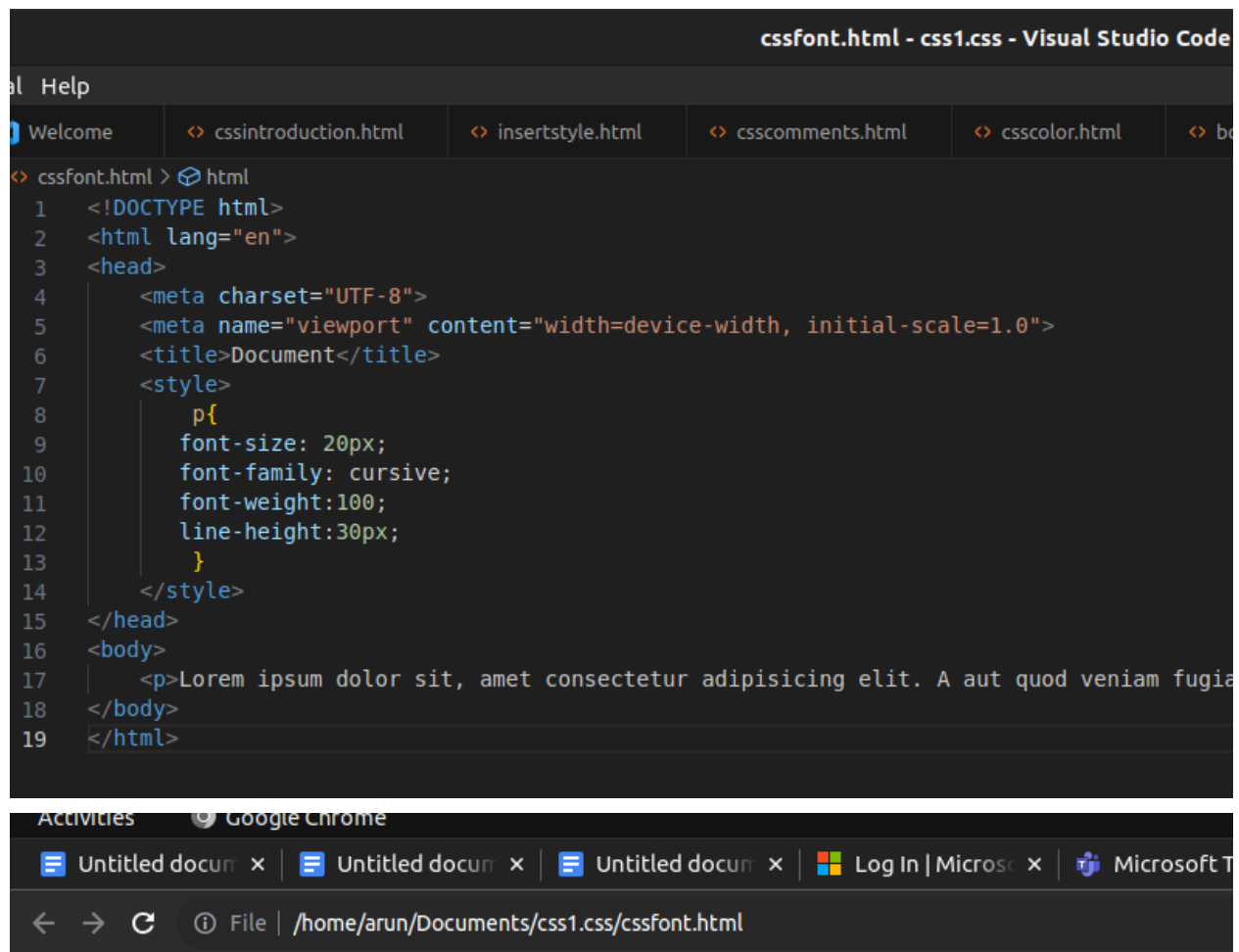
Relative size:

- Sets the size relative to surrounding elements
- Allows a user to change the text size in browsers

Font Google: If you do not want to use any of the standard fonts in HTML, you can use Google Fonts. Google Fonts are free to use, and have more than 1000 fonts to choose from. Just add a special style sheet link in the <head> section and then refer to the font in the CSS.

Font Shorthand: To shorten the code, it is also possible to specify all the individual font properties in one property. The font property is a shorthand property for:

- font-style
- font-variant
- font-weight
- font-size/line-height
- font-family



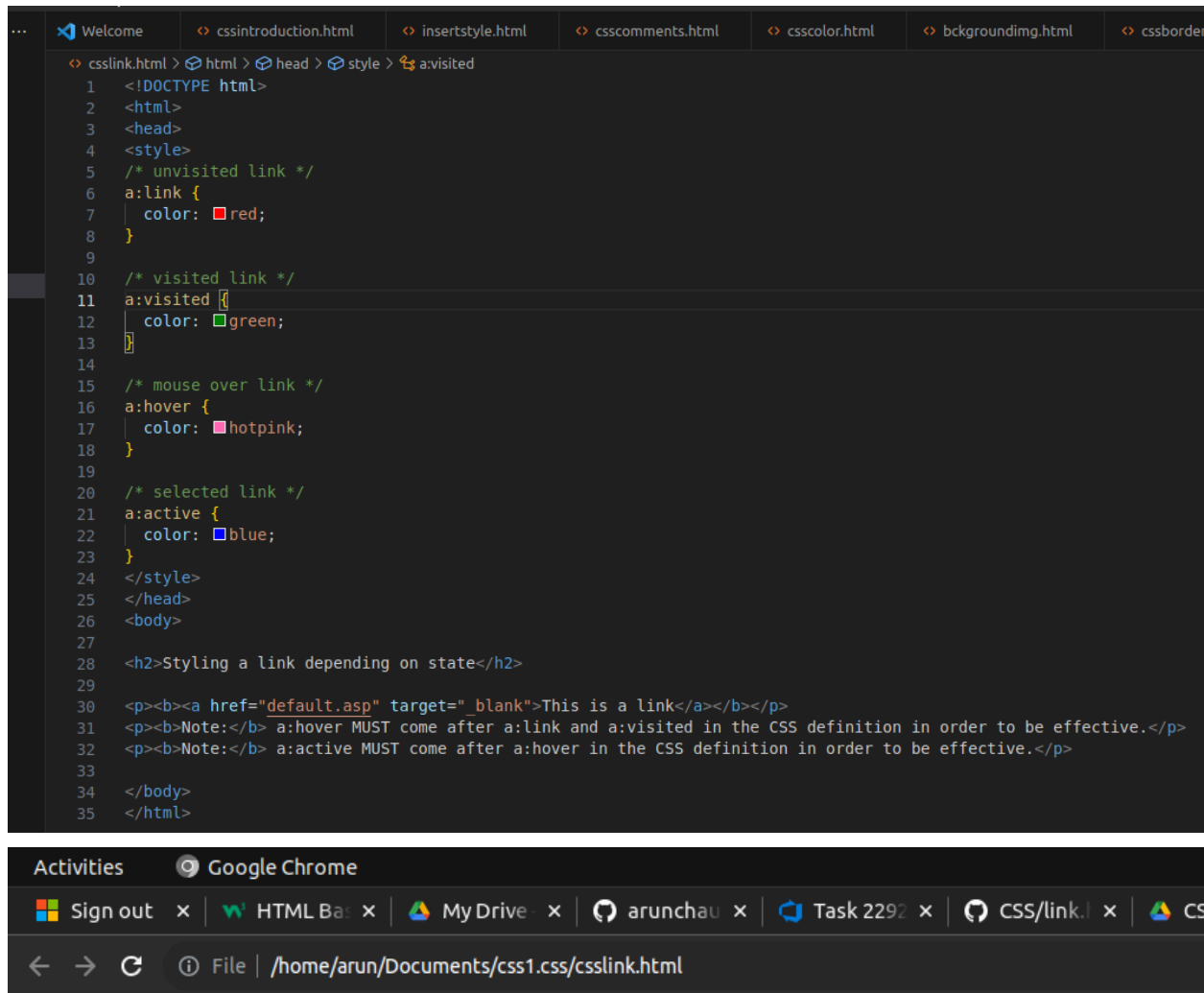
Lorem ipsum dolor sit, amet consectetur adipisicing elit. A aut quod veniam fugiat laudantium acc
maiores expedita quas a aspernatur unde perspiciatis! Deleniti repellendus quisquam rem modi ip

CSS Links

In addition, links can be styled differently depending on what state they are in.

The four links states are:

- a:link - a normal, unvisited link
- a:visited - a link the user has visited
- a:hover - a link when the user mouses over it
- a:active - a link the moment it is clicked



Styling a link depending on state

[This is a link](#)

Note: a:hover MUST come after a:link and a:visited in the CSS definition in order to be effective.

Note: a:active MUST come after a:hover in the CSS definition in order to be effective.

CSS Lists:

In HTML, there are two main types of lists:

- unordered lists () - the list items are marked with bullets
- ordered lists () - the list items are marked with numbers or letters

The CSS list properties allow you to:

- Set different list item markers for ordered lists
- Set different list item markers for unordered lists
- Set an image as the list item marker

```
...  <> cssintroduction.html  <> insertstyle.html  <> csscomments.html  <> csscolor.html  <> backgroundimg.html

<> csslist.html > html > body > ul
1  <!DOCTYPE html>
2  <html>
3  <head>
4  <style>
5  ul.demo {
6      list-style-type: none;
7      margin: 0;
8      padding: 0;
9  }
10 </style>
11 </head>
12 <body>
13
14 <p>Default list:</p>
15 <ul>
16     <li>Coffee</li>
17     <li>Tea</li>
18     <li>Coca Cola</li>
19 </ul>
20
21 <p>Remove bullets, margin and padding from list:</p>
22 <ul class="demo">
23     <li>Coffee</li>
24     <li>Tea</li>
25     <li>Coca Cola</li>
26 </ul>
27
28 </body>
29 </html>
```

```
<  >  ↻  ⓘ  File | /home/arun/Documents/css1.css/csslist.html
```

Default list:

- Coffee
- Tea
- Coca Cola

Remove bullets, margin and padding from list:

Coffee
Tea
Coca Cola

CSS Display:

The display property is the most important CSS property for controlling layout. The display property specifies if/how an element is displayed.

Every HTML element has a default display value depending on what type of element it is. The default display value for most elements is block or inline.

Block-level Elements:

A block-level element always starts on a new line and takes up the full width available (stretches out to the left and right as far as it can). The `<div>` element is a block-level element.

Examples of block-level elements:

- `<div>`
- `<h1>` - `<h6>`
- `<p>`
- `<form>`
- `<header>`
- `<footer>`
- `<section>`

Inline Elements:

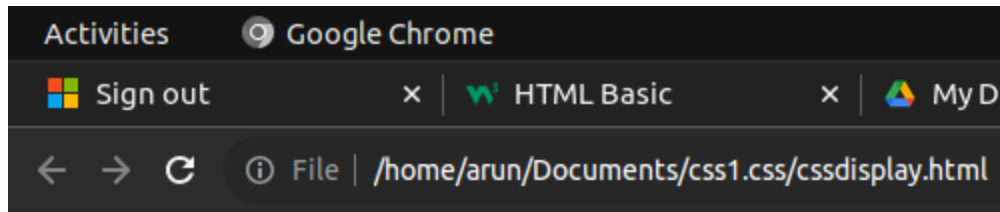
An inline element does not start on a new line and only takes up as much width as necessary. This is an inline `` element inside a Paragraph.

Examples of inline elements:

- ``
- `<a>`
- ``

cssdisplay.html > html > head > style > span

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7      <style>
8          *{
9              margin:0;
10             padding:0;
11             box-sizing:border-box;
12         }
13         .box{
14             width:200px;
15             height:200px;
16             margin:100px;
17             border:5px;
18             border-style: solid;
19             border-color: black;
20         }
21     }
22     span{
23         border:2px solid green;
24         width:100px;
25         height:100px;
26         margin:10px;
27         display:inline-block;
28         padding:10px;
29     }
30
31 </style>
32 </head>
33 <body>
34     <p class="box">Lorem, ipsum dolor sit amet consectetur adipisicing elit. Numquam volupt
35 <span>this is Arun</span>
36 <span>this is Nura</span>
37 </body>
38 </html>
```



Lorem, ipsum dolor sit amet
consectetur adipisicing elit.
Numquam voluptatibus
officiis incidunt quae non
consectetur nemo explicabo
odio placeat voluptate.

this
is
Arun

this
is
Nura

CSS Position: The position property specifies the type of positioning method used for an element (static, relative, fixed, absolute or sticky). The position property specifies the type of positioning method used for

an element.

There are five different position values:

- static
- relative
- fixed

- absolute
- sticky

Elements are then positioned using the top, bottom, left, and right properties. However, these properties will not work unless the position property is set first. They also work differently depending on the position value.

```
csscomments.html  csscolor.html  bckgroundimg.html  cssborder.html  cssboxmc
csspositions.html > html > body > div.container > img.image1
5  <meta name="viewport" content="width=device-width, initial-scale=1.0">
6  <title>Document</title>
7  <style>
8      div{
9          border:1px solid black;
10     }
11     img{
12         margin:10px;
13     }
14     .image2{
15         position:absolute;
16         top:100px;
17     }
18     .container{
19         position:relative;
20         height:8000px;
21     }
22     .image4{
23         position:fixed;
24     }
25     .image3{
26         position:sticky;
27         top:100px;
28     }
29     .box{
30         height:5000px;
31     }
32
33 </style>
34 </head>
35 <body>
36     <div class="container">
37         
41         
45         
49         
53     </div>
54     <div class="box">this is Arun</div>
```



CSS Overflow

The overflow property specifies whether to clip the content or to add scrollbars when the content of an element is too big to fit in the specified Area.

The overflow property has the following values:

- visible - Default. The overflow is not clipped. The content renders outside the element's box
- hidden - The overflow is clipped, and the rest of the content will be invisible
- scroll - The overflow is clipped, and a scrollbar is added to see the rest of the content
- auto - Similar to scroll, but it adds scrollbars only when Necessary


```
csscolor.html  backgroundimg.html  cssborder.html  cssboxmodel.html  cssfont.html  csslink.html  css...
cssoverflow.html > html > head > style > #img2
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7      <style>
8          .container{
9              border:1px solid black;
10             overflow:scroll;
11             height:500px;
12         }
13
14         #img1{
15             float:left;
16         }
17         #img2{
18             float:left;
19         }
20         #img3{
21             float:left;
22         }
23         .item{
24             width:200px;
25             height:200px;
26             margin:2px;
27             background-color:aquamarine;
28             border:1px red;
29         }
30     </style>
31 </head>
32 <body>
33     <div class="container">
34         
35         
36
37         
38     <p>Lorem ipsum dolor sit amet consectetur, adipisicing elit. Sequi eos, delectus voluptates animi ut con
39
40 </div>
41 </body>
42 </html>
```



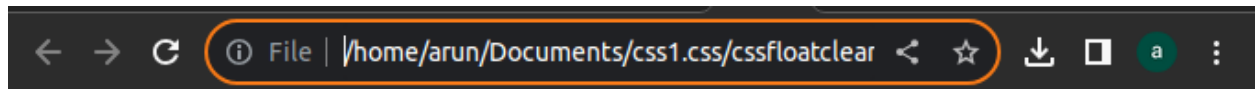
Float: The float property is used for positioning and formatting content e.g. let an image float left to the text in a container. The float property can have one of the following values:

- left - The element floats to the left of its container
- right - The element floats to the right of its container
- none - The element does not float (will be displayed just where it occurs in the text). This is default
- inherit - The element inherits the float value of its parent

Clear: When we use the float property, and we want the next element below (not on right or left), we will have to use the clear property. The clear property specifies what should happen with the element that is next to a floating element. The clear property can have one of the following Values:

- none - The element is not pushed below left or right floated elements. This is default
- left - The element is pushed below left floated elements
- right - The element is pushed below right floated elements
- both - The element is pushed below both left and right floated elements
- inherit - The element inherits the clear value from its parent

```
cssfloatclear.html > html > head > style > #img2
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7   <style>
8     #img1{
9       float:right;
10      height:250px;
11    }
12    #img2{
13      float:right;
14      height:250px;
15      clear:right;
16    }
17  </style>
18 </head>
19 <body>
20   <div class="container">
21     
22     
23
24     <p>Lorem, ipsum dolor sit amet consectetur adipisicing elit. Tenetur esse fuga dolor, eius accusantium labore adi
25     <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Reprehenderit aspernatur veritatis maxime vitae hic,
26
27   </div>
28 </body>
29 </html>
```



Lorem, ipsum dolor sit amet consectetur adipisicing elit. Tenetur esse fuga dolor, eius accusantium labore adipisci sed, eveniet aliquid soluta, nam veritatis. A obcaecati ex ratione nesciunt? Odit quasi dolorem tempora reiciendis repudiandae odio numquam iusto nemo architecto animi modi sint ducimus, officia perferendis unde, assumenda ea atque rem sunt!

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Reprehenderit aspernatur veritatis maxime vitae hic, fugit consequuntur natus, omnis ad harum sequi suscipit error dolorum. Ullam maiores rerum quibusdam distinctio provident eveniet corrupti, fugiat architecto tempore, nihil reprehenderit. Nostrum blanditiis doloribus praesentium cupiditate temporibus, error quae dolore ex saepe eaque tenetur, dicta dolorem eveniet quod aperiam natus velit quasi odit in expedita quaerat ipsam repellendus. Veniam dolore tempora veritatis, beatae repellendus perspiciatis neque earum, possimus voluptatum voluptates reprehenderit voluptatem quas, repudiandae amet quidem assumenda laboriosam sint error provident obcaecati modi. Aperiam facilis eum sequi consequatur eveniet consectetur aliquam, expedita minima aut?



CSS UNIT-

CSS units are used to specify measurements for various properties in

Cascading Style Sheets (CSS). They determine how elements are sized and positioned on a web page. Here are some commonly used CSS Units:

1. Pixels (px): A pixel is the smallest unit of measurement in CSS. It represents a single dot on a screen. Pixel values are fixed and do not change with the size of the viewport.

2. Percentages (%): Percentages are relative units that are based on the size of the parent element or the viewport. For example, setting the width of an element to 50% will make it take up half of its parent container's width.

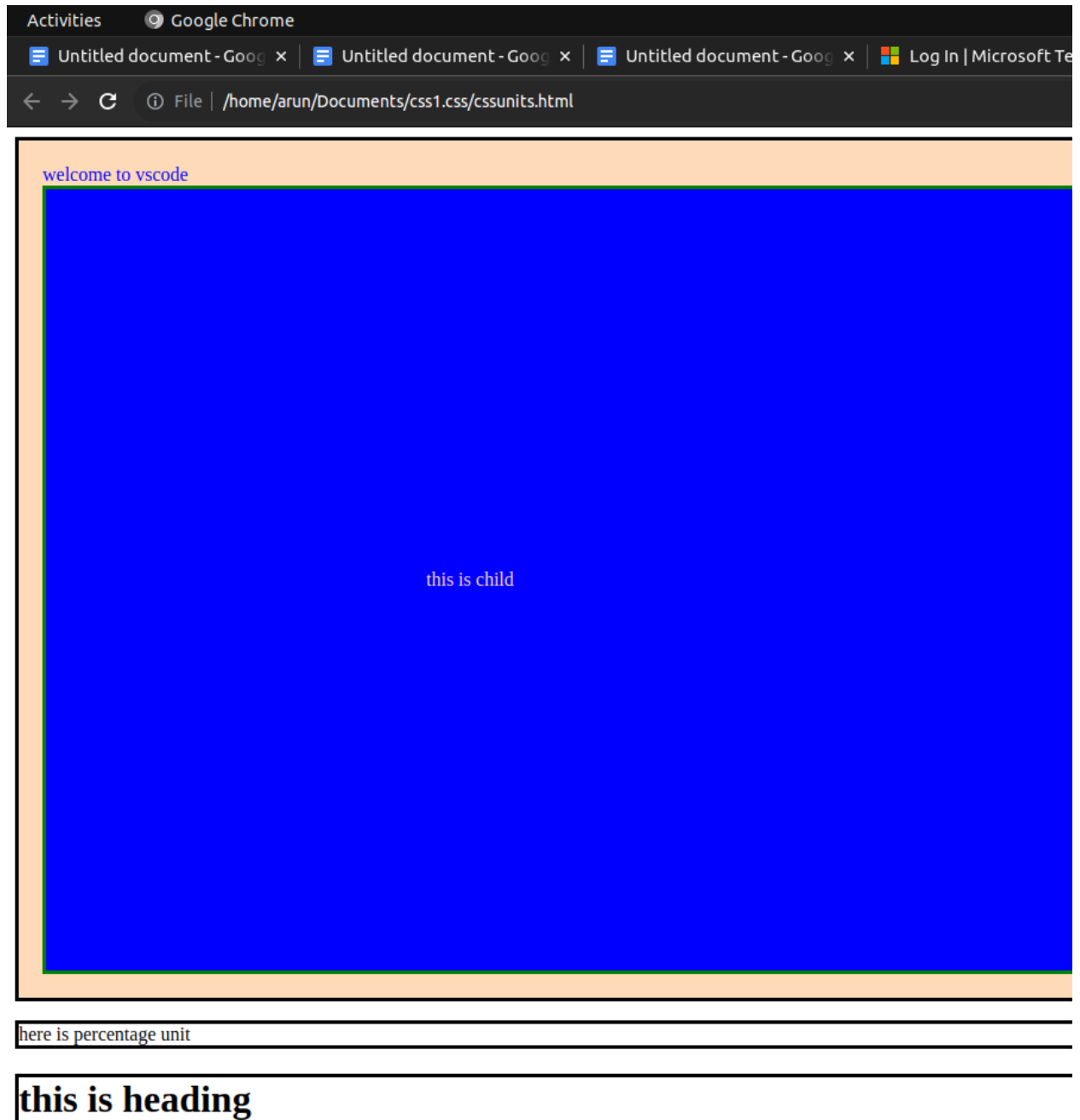
3. Em (em): The "em" unit is relative to the font size of the element or its parent. For example, if the font size of an element is 16 pixels, setting the margin to 1em will make the margin equal to 16 pixels.

4. Rem (rem): Similar to the "em" unit, the "rem" unit is relative to the root element's font size. It provides a way to establish a baseline font size for the entire page. Unlike "em," which is relative to the nearest parent element with a defined font size, "rem" is always relative to the root element.

5. Viewport Units: These units are relative to the size of the browser viewport. There are four viewport units available:

- vw: Represents 1% of the viewport width.
- vh: Represents 1% of the viewport height.
- vmin: Represents the smaller of vw or vh.
- vmax: Represents the larger of vw or vh.

```
cssunits.html > html > head > style > .box
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7   <style>
8     .box{
9       border-style:solid;
10      border-color:black;
11      color:blue;
12      background-color:peachpuff;
13      padding:20px;
14    }
15    .child{
16      border-style:solid;
17      border-color:green;
18      background-color:blue;
19      color:peachpuff;
20      padding:20em;
21    }
22    p{
23      border-style: solid;
24      border-color:black;
25      width:50%;
26    }
27    h1{
28      border-style: solid;
29      border-color:black;
30      width:75vw;
31    }
32  </style>
33 </head>
34 <body>
35   <div class="box">
36     welcome to vscode
37     <div class="child">
38       this is child
39     </div>
40   </div>
41   <p>
42     here is percentage unit
43   </p>
44   <h1>this is heading</h1>
45 </body>
46 </html>
47 </html>
```



CSS Rounded Corners

With the CSS `border-radius` property, you can give any element "rounded corners".

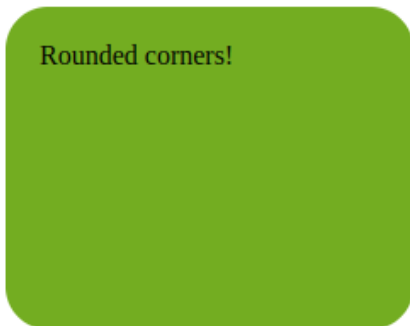
<> cssroundcorners.html > head > style > #rcorners3

```
1  html>
2  <head>
3  <style>
4  #rcorners1 {
5      border-radius: 25px;
6      background: #73AD21;
7      padding: 20px;
8      width: 200px;
9      height: 150px;
10 }
11
12 #rcorners2 {
13     border-radius: 25px;
14     border: 2px solid #73AD21;
15     padding: 20px;
16     width: 200px;
17     height: 150px;
18 }
19
20 #rcorners3 {
21     border-radius: 25px;
22     background: url(paper.gif);
23     background-position: left top;
24     background-repeat: repeat;
25     padding: 20px;
26     width: 200px;
27     height: 150px;
28 }
29 </style>
30 </head>
31 <body>
32
33 <h1>The border-radius Property</h1>
34
35 <p>Rounded corners for an element with a specified background color:</p>
36 <p id="rcorners1">Rounded corners!</p>
37 <p>Rounded corners for an element with a border:</p>
38 <p id="rcorners2">Rounded corners!</p>
39 <p>Rounded corners for an element with a background image:</p>
40 <p id="rcorners3">Rounded corners!</p>
41
42 </body>
43 </html>
```

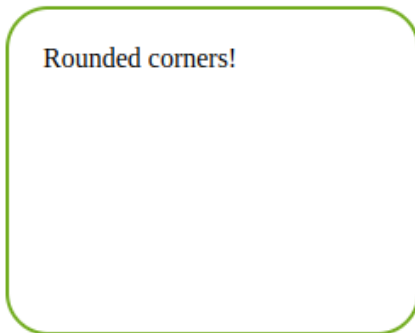
html>

The border-radius Property

Rounded corners for an element with a specified background color:



Rounded corners for an element with a border:



Rounded corners for an element with a background image:

Rounded corners!

CSS Shadow Effects

With CSS you can add shadow to text and to elements.

In these chapters you will learn about the following properties:

- text-shadow
- box-shadow

CSS Text Shadow

The CSS text-shadow property applies shadow to text.

CSS box-shadow Property

The CSS box-shadow property is used to apply one or more shadows to an Element.

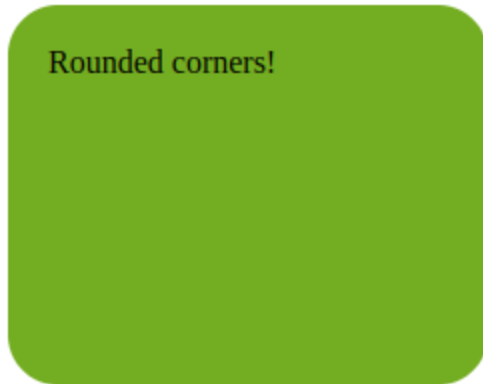
[cssboxmodel.html](#)[cssfont.html](#)[csslink.html](#)[csslist.html](#)[cssdisplay.html](#)[cssroundcorners.html](#) > head > style

```
1  html>
2  <head>
3  <style>
4  #rcorners1 {
5      border-radius: 25px;
6      background: #73AD21;
7      padding: 20px;
8      width: 200px;
9      height: 150px;
10 }
11
12 #rcorners2 {
13     border-radius: 25px;
14     border: 2px solid #73AD21;
15     padding: 20px;
16     width: 200px;
17     height: 150px;
18 }
19
20 #rcorners3 {
21     border-radius: 25px;
22     background: url(paper.gif);
23     background-position: left top;
24     background-repeat: repeat;
25     padding: 20px;
26     width: 200px;
27     height: 150px;
28 }
29 </style>
30 </head>
31 <body>
32
33 <h1>The border-radius Property</h1>
34
35 <p>Rounded corners for an element with a specified background color:</p>
36 <p id="rcorners1">Rounded corners!</p>
37 <p>Rounded corners for an element with a border:</p>
38 <p id="rcorners2">Rounded corners!</p>
39 <p>Rounded corners for an element with a background image:</p>
40 <p id="rcorners3">Rounded corners!</p>
41
42 </body>
43 </html>
```

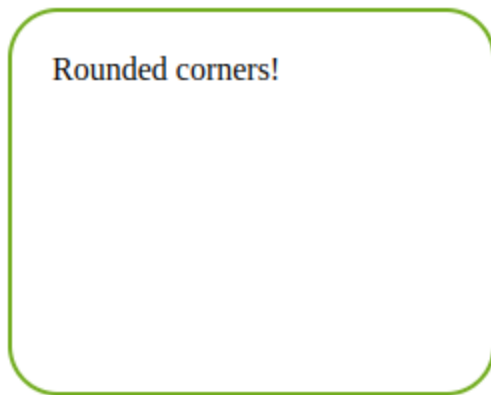
html>

The border-radius Property

Rounded corners for an element with a specified background color:



Rounded corners for an element with a border:



Rounded corners for an element with a background image:

Rounded corners!

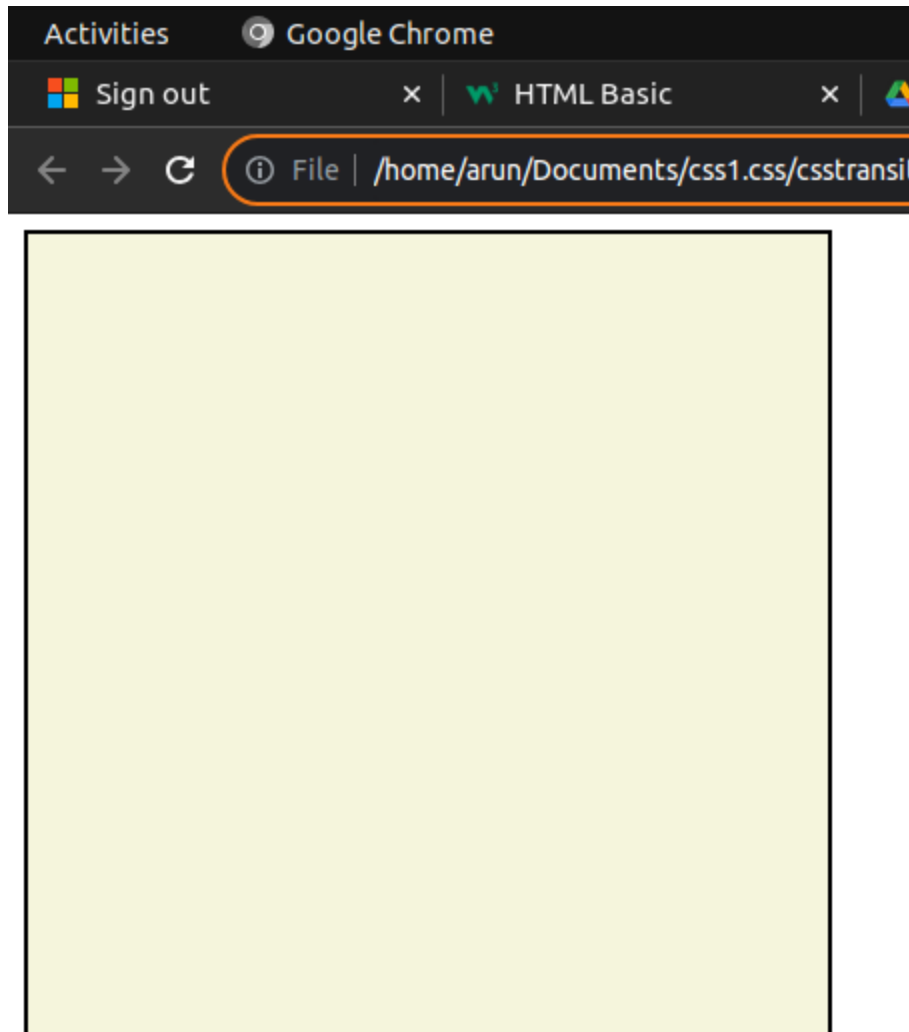
CSS Transitions

CSS transitions allows you to change property values smoothly, over a given duration.

In this chapter you will learn about the following properties:

- transition
- transition-delay
- transition-duration
- transition-property
- transition-timing-function

```
cssFont.html  cssLink.html  cssList.html  cssDisplay.html  cssPositions.html
cssTransitions.html > html > head > style > .box:hover
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7      <style>
8          .box{
9              height:400px;
10             width:400px;
11             background-color: ■ beige;
12             border:2px solid □ black;
13             transition-property: all;
14             transition-duration: 4s;
15             transition-delay:1s;
16             transition-timing-function:linear;
17         }
18         .box:hover{
19             width:1200px;
20             height: 900px;
21             background-color: ■ red;
22         }
23     }
24 </style>
25 </head>
26 <body>
27     <div class="box">
28
29     </div>
30 </body>
31 </html>
```



CSS Animations

CSS allows animation of HTML elements without using JavaScript or Flash!

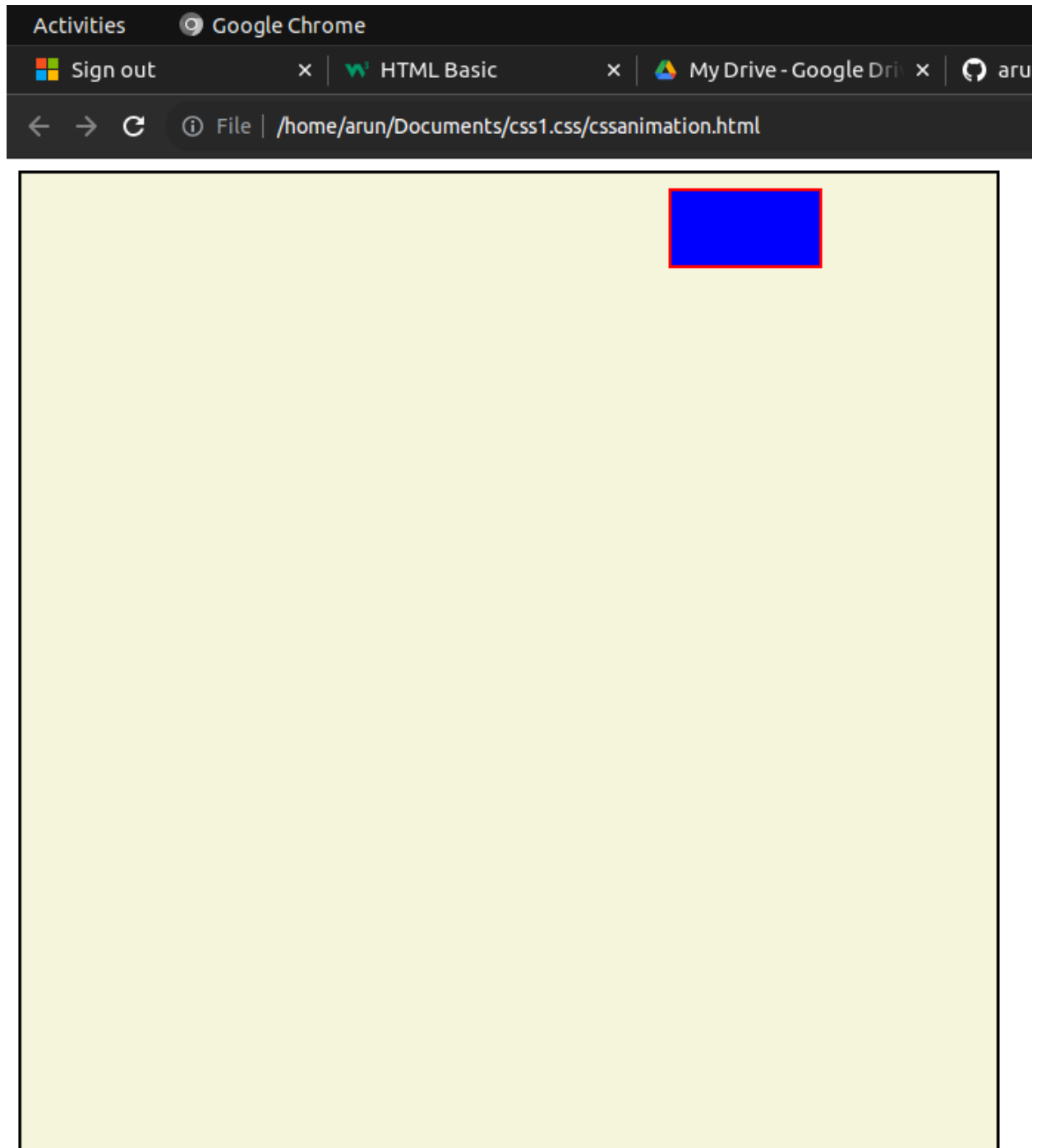
CSS

In this chapter you will learn about the following properties:

- @keyframes
- animation-name
- animation-duration
- animation-delay
- animation-iteration-count
- animation-direction

- animation-timing-function
- animation-fill-mode
- animation

```
csslink.html < csslist.html < cssdisplay.html < csspositions.html < cssoverflow.html < cssfloatclear.h
cssanimation.html > html > head > style
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7
8   <style>
9     .container{
10       background-color: ■ beige;
11       border:2px solid □ black;
12       height:660px;
13       width:660px;
14       display:flex;
15       justify-content:start;
16       align-items:start;
17
18     }
19
20     .box{
21       background-color: ■ blue;
22       height:50px;
23       width:100px;
24       border:2px solid ■ red;
25       margin:10px;
26       position:relative;
27       animation: 5s linear 1s infinite alternate none running rightMovement;
28     }
29
30     @keyframes rightMovement{
31     from{
32       top:0;
33       left:0;
34     }
35     to{
36       top:0;
37       left:1250px;
38     }
39     }
40
41   </style>
42 </head>
43 <body>
44   <div class="container">
45     <div class="box">
46     </div>
47   </div>
48 </body>
49 </html>
```



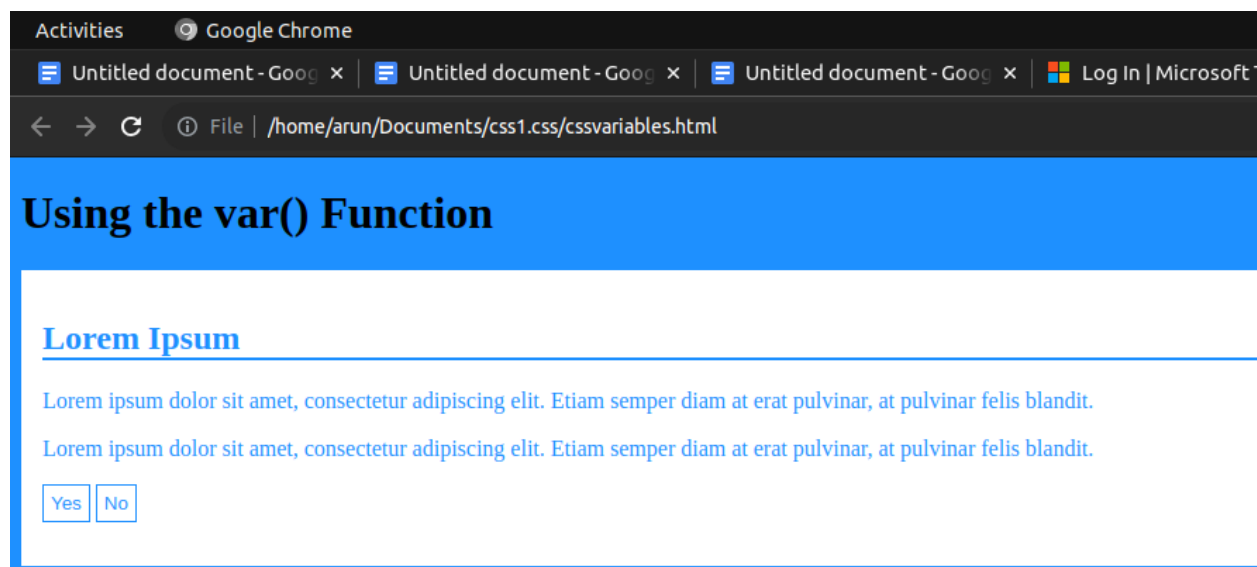
CSS Variables

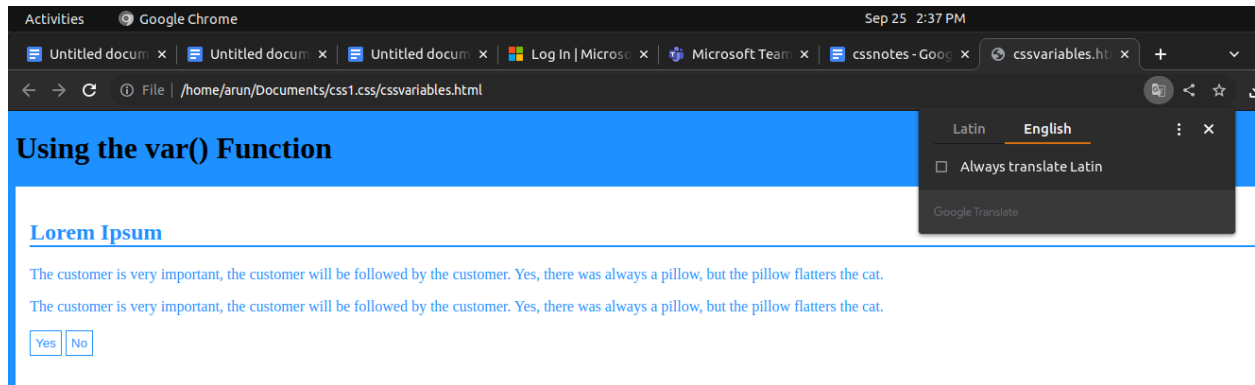
The `var()` function is used to insert the value of a CSS variable.

CSS variables have access to the DOM, which means that you can

create variables with local or global scope, change the variables with JavaScript, and change the variables based on media queries.

```
html > cssdisplay.html > csspositions.html > cssoverflow.html > cssfloatclear.html > cssunits.html > cssroundcorners.html > csstransitions.html
cssvariables.html > html > head > style > :root
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <style>
5 :root {
6   --blue: #1e90ff;
7   --white: #ffffff;
8 }
9
10 body {
11   background-color: var(--blue);
12 }
13
14 h2 {
15   border-bottom: 2px solid var(--blue);
16 }
17
18 .container {
19   color: var(--blue);
20   background-color: var(--white);
21   padding: 15px;
22 }
23
24 button {
25   background-color: var(--white);
26   color: var(--blue);
27   border: 1px solid var(--blue);
28   padding: 5px;
29 }
30 </style>
31 </head>
32 <body>
33
34 <h1>Using the var() Function</h1>
35
36 <div class="container">
37   <h2>Lorem Ipsum</h2>
38   <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit.</p>
39   <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit.</p>
40   <p>
41     <button>Yes</button>
42     <button>No</button>
43   </p>
44 </div>
45
46 </body>
47 </html>
```





CSS3 Introduced Media Queries

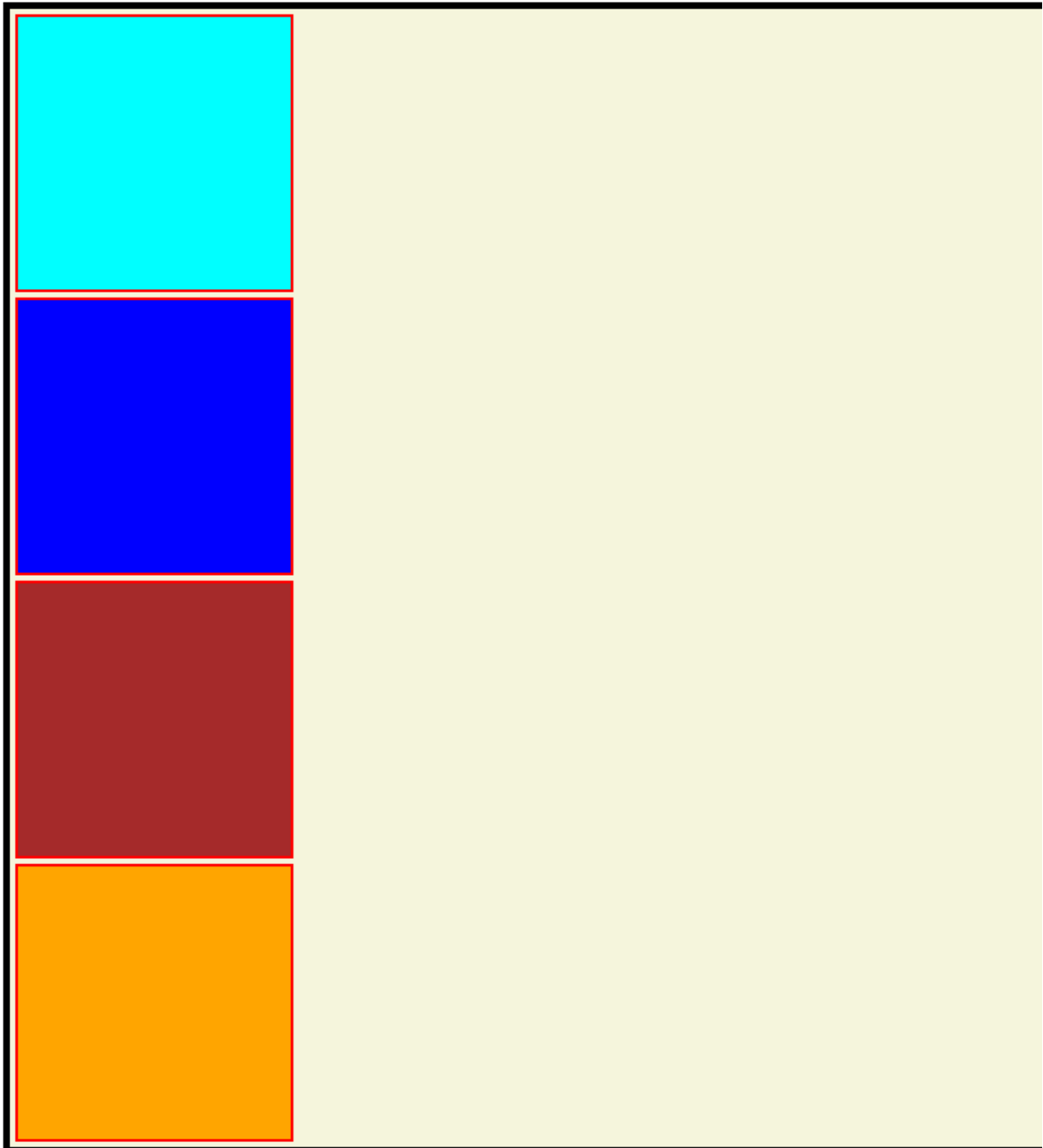
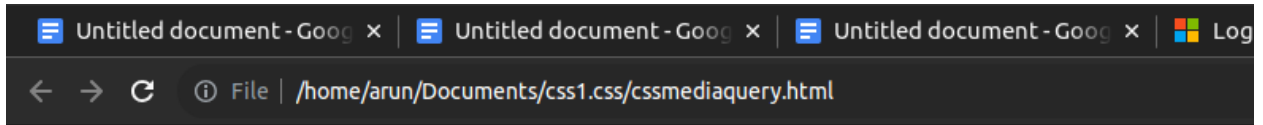
Media queries in CSS3 extended the CSS2 media types idea: Instead of looking for a type of device, they look at the capability of the device.

Media queries can be used to check many things, such as:

- width and height of the viewport
- width and height of the device
- orientation (is the tablet/phone in landscape or portrait mode?)
- resolution

Using media queries are a popular technique for delivering a tailored style sheet to desktops, laptops, tablets, and mobile phones (such as iPhone and Android phones).

```
cssmediaquery.html > html > head > style > .box
7   <style>
8   .container{
9     background-color: beige;
10    border:5px solid black;
11    margin:2px;
12    padding:2px;
13    display:flex;
14  }
15  .box{
16    height:200px;
17    width:200px;
18    border:2px solid red;
19    margin:2px;
20    padding:2px;
21  }
22  #box1{
23    background-color: aqua;
24    /*flex-shrink:4;*/
25  }
26  #box2{
27    background-color: blue;
28    /*flex-grow:1;*/
29  }
30  #box3{
31    background-color: brown;
32  }
33  #box4{
34    background-color: orange;
35  }
36  @media (min-width:640px)
37  {
38    .container{
39      flex-direction:column;
40    }
41  }
42  @media (max-width:640px)
43  {
44    .container{
45      flex-direction: row;
46    }
47  }
48  </style>
49  </head>
50  <body>
51    <div class="container">
52      <div class="box" id="box1"></div>
53      <div class="box" id="box2"></div>
54      <div class="box" id="box3"></div>
55      <div class="box" id="box4"></div>
56    </div>
57  </body>
```



CSS Flexbox

It is a powerful tool for creating layout.

CSS Flex Container

The flex container properties are:

- flex-direction
- flex-wrap
- flex-flow
- justify-content
- align-items
- align-content

The flex-direction property accepts four possible values:

1. row (default): This value arranges the flex items horizontally in a row from left to right. The main axis is horizontal, and the cross axis is vertical.
2. row-reverse: This value arranges the flex items horizontally in a row from right to left. The main axis is still horizontal, but the order of the flex items is reversed compared to row.
3. column: This value stacks the flex items vertically from top to bottom. The main axis is vertical, and the cross axis is horizontal.
4. column-reverse: This value stacks the flex items vertically from bottom to top.

The flex-wrap property accepts four possible values:

- 1-wrap
- 2-nowrap
- 3-wrap-reverse

The justify-content Property

The justify-content property in CSS is used to align flex items along the main axis of a flex container. It determines how extra space is distributed within the flex container when the flex items do not fill the entire line.

The justify-content property accepts several values:

1. flex-start (default): This value aligns the flex items at the start of the flex container. The items are packed towards the beginning of the main axis.
2. flex-end: This value aligns the flex items at the end of the flex container. The items are packed towards the end of the main axis.
3. center: This value aligns the flex items at the center of the flex container. The items are evenly distributed, with equal space before and after them.
4. space-between: This value evenly distributes the flex items along the main axis. The first item is aligned to the start of the flex container, the last item is aligned to the end, and the remaining items are evenly spaced in between.
5. space-around: This value evenly distributes the flex items along the main axis, but with space both before and after each item. The space before the first item and after the last item is half the size of the space between the items.
6. space-evenly: This value evenly distributes the flex items along the main axis with equal space before, between, and after each item.

The align-items Property

The align-items property in CSS is used to align flex items along the cross-axis of a flex container. It controls the vertical alignment of flex items when they do not fill the entire height of the flex container.

The align-items property accepts several values:

1. stretch (default): This value stretches the flex items to fill the height of the flex container along the cross-axis. This is the default behavior if no align-items value is specified.
2. flex-start: This value aligns the flex items at the start of the flex container along the cross-axis. The items are aligned at the top of the container.
3. flex-end: This value aligns the flex items at the end of the flex container along the cross-axis. The items are aligned at the bottom of the container.
4. center: This value aligns the flex items at the center of the flex

container along the cross-axis.

Align-items- it is used to decrease space between items.

CSS Flex Items

Here are some commonly used properties for flex items:

1. flex-grow: This property specifies how flex items should grow to fill the available space along the main axis. It accepts a numeric value, which represents the proportion of available space a flex item should take up compared to other flex items. For example, a value of 1 will make a flex item grow to take up the same amount of space as other flex items with a value of 1.

2. flex-shrink: This property specifies how flex items should shrink if the available space along the main axis is insufficient to accommodate all flex items. It also accepts a numeric value, and a higher value means a flex item will shrink more compared to other flex items.

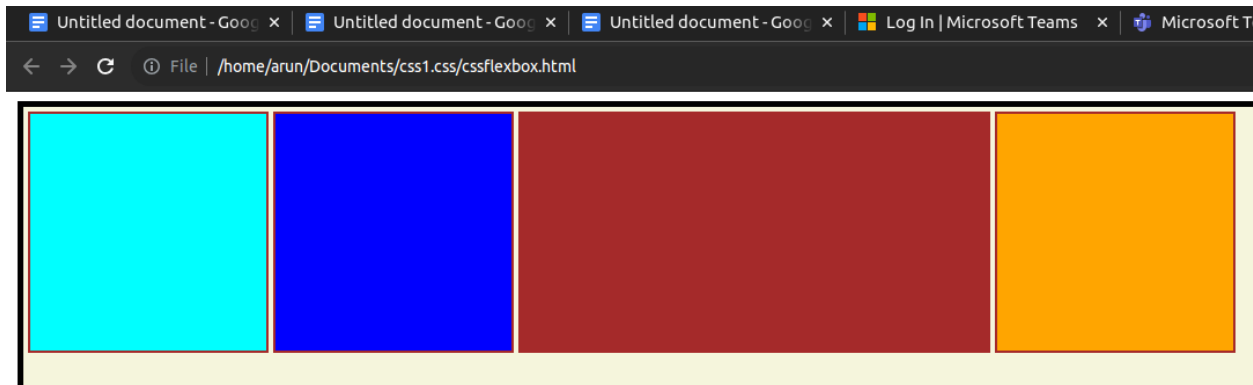
3. flex-basis: This property defines the initial size of a flex item along the main axis before any remaining space is distributed. It can be set to a length value like pixels or percentages, or the auto keyword to use the item's intrinsic size.

4. flex: This is a shorthand property that combines flex-grow, flex-shrink, and flex-basis in a single declaration. It accepts three values separated by spaces. For example, flex: 1 0 auto; sets the flex-grow to 1, flex-shrink to 0, and flex-basis to auto.

5. order: This property allows you to control the order in which flex items appear within the flex container. By default, flex items have an order of 0, but you can assign positive or negative integer values to rearrange their order.

6. align-self: This property allows you to override the align-items property for an individual flex item. It controls the alignment of the specific item along the cross-axis. The available values are the same as align-items, such as flex-start, flex-end, center, baseline, and stretch.

```
nl  < cssoverflow.html  < cssfloatclear.html  < cssunits.html  < cssroundcorners.html  < csstransit
< cssflexbox.html > html > head > style > .container
5  <meta name="viewport" content="width=device-width, initial-scale=1.0">
6  <title>Document</title>
7  <style>
8      .container{
9          background-color: ■ beige;
10         border:5px solid □ black;
11         margin:2px;
12         padding:2px;
13         display:flex;
14         height:800px;
15         justify-content:start;
16         align-items:start;
17
18
19     }
20     .box{
21         height:200px;
22         width:200px;
23         border:2px solid ■ brown;
24         margin:2px;
25         padding:2px;
26
27     }
28     #box1{
29         background-color: ■ aqua;
30         /*flex-shrink:4;*/
31     }
32     #box2{
33         background-color: ■ blue;
34         /*flex-grow:1;*/
35     }
36     #box3{
37         background-color: ■ brown;
38         flex-basis:400px;
39     }
40     #box4{
41         background-color: ■ orange;
42     }
43
44 </style>
45 </head>
46 <body>
47     <div class="container">
48         <div class="box" id="box1"></div>
49         <div class="box" id="box2"></div>
50         <div class="box" id="box3"></div>
51         <div class="box" id="box4"></div>
52     </div>
53 </body>
54 </html>
```

Grid Layout

The CSS Grid Layout Module offers a grid-based layout system, with rows and columns, making it easier to design web pages without having to use floats and positioning.

Here are some key points about CSS Grid:

1. Grid Container: To create a grid layout, you define a grid container by applying the `display: grid;` property to an element.

This makes the element a grid container, and its direct children become grid items.

2. Grid Tracks: CSS Grid consists of horizontal and vertical tracks that form the grid's rows and columns. You can define the size of these tracks using various length units like pixels, percentages, or Auto.

3. Grid Lines: The grid lines separate the tracks and define the boundaries for placing grid items. They can be numbered using line numbers or named using custom names that you assign.

4. Grid Rows and Columns: You can specify the number and size of rows and columns using properties like `grid-template-rows` and `grid-template-columns`. You can also use shorthand properties like `grid-template` to define both rows and columns in a single Declaration.

5. Grid Placement: Grid items can be placed explicitly using line numbers or names of grid lines, or implicitly by allowing them to

flow into the grid according to the default placement behavior.

6. Grid Areas: You can define named grid areas using the `grid-template-areas` property. By assigning specific names to areas within the grid, you can easily position and place grid items in those areas.

7. Grid Alignment: CSS Grid provides powerful alignment properties for positioning grid items within grid cells. Properties like `justify-self` and `align-self` allow you to align items horizontally and vertically within their respective cells.

8. Grid Responsiveness: CSS Grid offers built-in features for creating responsive layouts. You can use media queries, fractional units, or the `auto-fill` and `auto-fit` keywords to adapt the grid layout based on different screen sizes.

```
... .html  cssfloatclear.html  cssunits.html  cssroundcorners.html  csstransitions.html  cssanimation.html  cssvariables.html
cssgrids.html > html > head > style > main
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5    <meta name="viewport" content="width=device-width, initial-scale=1.0">
6    <title>Document</title>
7    <style>
8      main{
9        width:90%;
10       margin:50px;
11       border:1px solid black;
12       color:white;
13       display:grid;
14       grid-template-rows: 150px 150px;
15
16       grid-template-columns:repeat(auto-fit,minmax(150px,1fr));
17
18
19       /*grid-row-gap:20px;
20       grid-column-gap:20px;
21       grid-template-columns:min-content repeat(3,1fr);*/
22       /*grid-gap:20px;
23       grid-auto-rows:150px;
24       align-items:center;*/
25     }
26
27     .div1{
28       background-color: yellow;
29     }
30     .div2{
31       background-color: red;
32     }
33
34   }
35   .div3{
36     background-color: green;
37   }
38   .div4{
39     background-color: black;
40   }
41   .div5{
42     background-color: orange;
43     grid-row-start:1; grid-row-end:2;
44     /*grid-row:1/2;
45     grid-column:2/4;*/
46
47   }
48 }
49 .div6{
50   background-color: aqua;
51 }
52 </style>
53 </head>
54 <body>
55   <main>
56     <div class="div1">div1</div>
57     <div class="div2">div2</div>
58     <div class="div3">div3</div>
59     <div class="div4">div4</div>
60     <div class="div5">div5</div>
61     <div class="div6">div6</div>
62   </main>
63
64
65
66
67
68 </body>
69 </html>
```



CSS 2D Transforms

CSS transforms allow you to move, rotate, scale, and skew elements. With the CSS transform property you can use the following 2D transformation methods:

- `translate()`
- `rotate()`
- `scaleX()`
- `scaleY()`
- `scale()`
- `skewX()`
- `skewY()`
- `skew()`
- `matrix()`

The `translate()` method moves an element from its current position (according to the parameters given for the X-axis and the Y-axis).

the `rotate()` method rotates an element clockwise or counter-clockwise according to a given degree.

The `scale()` method increases or decreases the size of an element (according to the parameters given for the width and height).

The `skew()` method skews an element along the X and Y-axis by the given angles.

```
nl  < cssunits.html  < cssroundcorners.html  < csstransitions.html  < cssanimation.html  < cssvariables.f
< css2Dtransform.html > html > head > style > .box:hover
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7      <style>
8          .container{
9              display:flex;
10             justify-content: center;
11             height:100vh;
12             width:100vw;
13             align-items:center;
14             margin:auto;
15         }
16         .box{
17             height:400px;
18             width:400px;
19             border: 2px solid black;
20             background-color: beige;
21             display:flex;
22             justify-content: center;
23             align-content:center;
24             margin:auto;
25             transition:all 3s linear 1s;
26         }
27
28     }
29     .box:hover{
30         /*transform:rotate(130deg);
31         transform:scale(2);
32         transform:translate(500px)
33         transform:skew(45deg)
34         transform:translate(0,500px)*/
35         transform:rotate(130deg);
36     }
37     p{
38         font-size:28px;
39     }
40
41
42 </style>
43 </head>
44 <body>
45     <div class="container">
46         <div class="box">
47             <p>
48                 this is a help box
49             </p>
50         </div>
51     </div>
```

this is a help box

