CSS

Using CSS (Cascading Style Sheets), you can apply styles to your web pages to make them look exactly how you want.

Styling

There are several ways to add styling to a web page:

1. Define a style for a HTML tag. The following defines red, 20pt, Arial for the <h1> tag. So whenever you use the <h1> tag, this style will be used.

```
<!DOCTYPE html>
<html>
   <head>
      <style>
         h1 { color: red; font-size:20pt; font-family:Arial; }
      </style>
   <head>

   <body>
      <h1>Hello world</h1>
   </body>
</html>
```

2. When you wish to style a whole site, rather than a single page, a better way to manage style sheets is to move them completely out of your web pages to separate CSS files, and then import the ones you need. This lets you apply different style sheets for different layouts, without changing the HTML. This also allows a unify theme for all the pages because changing a rule only needs to be done once in this one file. In the following example, the file **styles.css** containing all of the CSS rules is being imported.

```
<style>
   @import url('styles.css');
</style>
```

The following shows a sample of what a CSS file might contain.

```
body {
  background-color: lightblue;
}

h1 {
  color: white;
  text-align: center;
}

p {
  font-family: verdana;
  font-size: 20px;
}
```

3. Define a style individually for just one element.

```
<div style='font-style:italic; color:blue;'>Hello there</div>
```

4. Using IDs. A better way for setting the style of an element is to assign an ID to it in the HTML. This states that the contents of the <div> with the ID **welcome** should have applied to them the style defined in the **welcome** style setting. Note the use of the # symbol, which specifies that only the ID with the name **welcome** should be styled with this statement.

```
<!DOCTYPE html>
<html>
   <head>
      <style>
         #welcome { color: blue; font-style:italic; }
      </style>
   <head>
   <body>
      <div id='welcome'>Hello world</div>
   </body>
</html>
```

5. Using Classes. If you would like to apply the same style to many elements, you do not have to give each one a different ID because you can specify a class to manage them all. Instead of the # symbol, which is reserved for IDs, class statements are prefaced with a . (period).

```
<!DOCTYPE html>
<html>
   <head>
      <style>
         .welcome { color: green; font-style:italic; }
      </style>
   <head>
   <body>
      <div class='welcome'>Hello world</div>
   </body>
</html>
```

CSS Rule Syntax

The syntax for a CSS rule is

selector { *property*:*value*; }

For example,

h1 { font-size:240% }

Each statement in a CSS rule starts with a *selector*, which is the item to which the rule will be applied. In the above example, the rule is applied to the selector h1, so any contents inside the <h1> </h1> tags will have this rule applied. The property is font-size and the value is 240%.

You can have as many *property*:*value* pairs as you like inside the curly braces { and }. They can be on the same line or different lines.

CSS Selectors

The means by which you access one or more elements is called *selection*, and the part of a CSS rule that does this is known as a *selector*. There are many varieties of selector.

The Type Selector

The type selector works on types of HTML elements such as <p> or <i>. For example, the following rule will ensure that all text within <p> … </p> tags is fully justified.

```
p { text-align:justify; }
```

The Descendant Selector

Descendant selectors let you apply styles to elements that are contained within other elements. For example, the following rule sets all text within <b> … </b> tags to red, but only if they occur within <p> … </p> tags like <p><i><b>Hello</b></i></p>.

```
p b { color:red; }
```

The Child Selector

Similar to descendant but only if b is directly in p like <p><b>Hello</b></p>.

```
p > b { color:red; }
```

The ID Selector

```
#myid { text-decoration:underline; }
```

```
<div id='myid'>Hello</div>
```

IDs can be used only once so only the first occurrence found will receive the property value.

The Class Selector

```
.myclass { margin-left:10px; }
```

```
<div class='myclass'>Hello</div>
```

The Attribute Selector

Style all the elements with the attribute type="submit"

```
form input[type="submit"] { width:100px; }
```

Selecting by Group

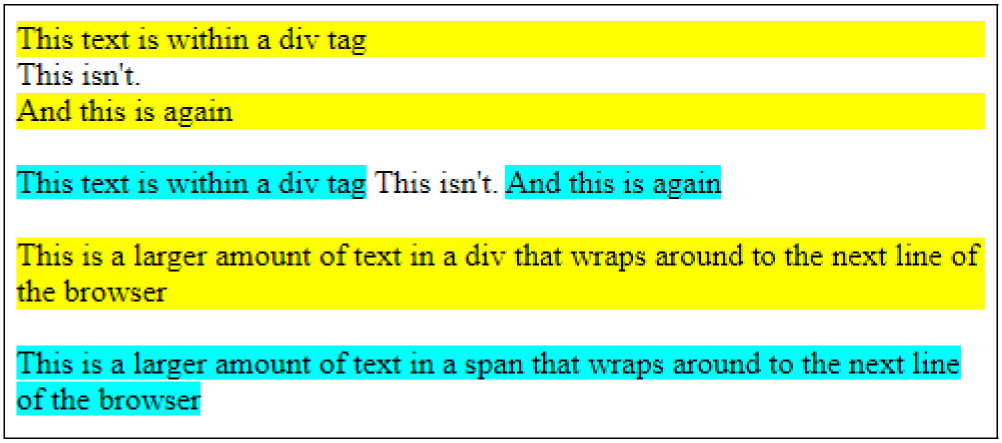
You can apply a rule to more than one element, class, or any other type of selector at the same time by separating the selectors with commas.

```
p, #idname, .classname { border-bottom:1px dotted orange; }
```

Difference between Div and Span Elements

By default, a <div> element goes all the way to the browser’s edge, whereas, a <span> element is only as wide as the text it contains.

```
<!DOCTYPE html>
<html>
   <head>
      <style>
         div, span { border:1px sold black; }
         div { background-color:yellow }
         span { background-color:cyan; }
      </style>
   <head>
   <body>
      <div>This text is within a div tag</div>
      This isn't.
      <div>And this is again</div>
      <br>
      <span>This text is within a div tag</span>
      This isn't.
      <span>And this is again</span>
      <br><br>
      <div>This is a larger amount of text in a div that wraps
around to the next line of the browser</div>
      <br>
      <span>This is a larger amount of text in a span that wraps
around to the next line of the browser</span>
   </body>
</html>
```



This text is within a div tag
This isn't.
And this is again

This text is within a div tag This isn't. And this is again

This is a larger amount of text in a div that wraps around to the next line of the browser

This is a larger amount of text in a span that wraps around to the next line of the browser

Measurements

- **Pixels**. A pixel is a single dot on the screen

```
.classname { margin:5px; }
```

- **Points**. A point is equal to 1/72 of an inch

```
.classname { font-size:14pt; }
```

- **Ems**. An em is equal to the current font size. So the provided number is relative to the current font size.

```
.classname { font-size:2em; }
```

- **Percent**. Relative to the current object size.

```
.classname { height:120%; }
```

Fonts and Typography

- **font-family.** List several so that if one is not available will fall back to the next one.

```
p { font-family:"Times New Roman", Georgia, serif; }
p { font-family:Arial, Verdana, Helvetica, sans-serif; }
```

- **font-size.**

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

```
p { font-size:140%; }
```

- **font-style.**

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

- **font-weight.**

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

- **font-decoration.**

```
p { font-decoration:underline; }
p { font- decoration:overline; }
p { font-decoration:line-through; }
p { font-decoration:blink; }
```

- **Alignment.**

```
p { text-align:left; }
p { text-align:right; }
p { text-align:center; }
p { text-align:justify; }
```

- **Transformation.**

```
p { text-transform:uppercase; }
p { text-transform:none; }
p { text-transform:capitalize; }
p { text-transform:lowercase; }
```

- **Indent.**

```
p { text-indent:20px; }
```

Colors

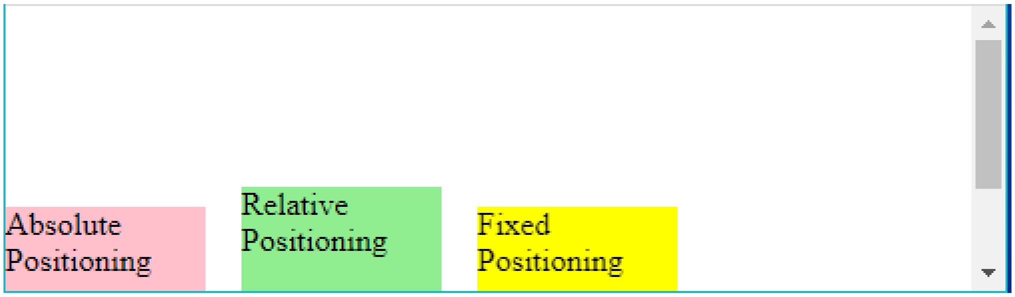
You can apply colors to the foreground and background using **color** and **background-color** respectively. The color value can either be a standard name or rgb value.

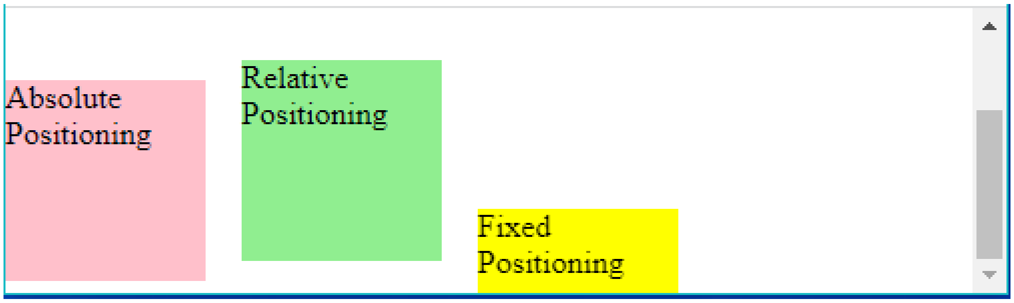
```
#object { color:silver; }
div { background-color:#ffff00; }
body { background-color:rgb(0,255,255); }
```

Positioning Elements

You can position an element using one of **absolute**, **relative**, or **fixed**, and **top**, **right**, **bottom**, and **left**. Both absolute and relative will scroll, whereas, fixed will not scroll.

```
<!DOCTYPE html>
<html>
   <head>
      <style>
         #object1 {
            position:absolute;
            background :pink;
            width    :100px;
            height     :100px;
            top        :100px;
            left     :0px;
            }
         #object2 {
            position:relative;
            background :lightgreen;
            width    :100px;
            height     :100px;
            top        :-8px;
            left     :110px;
            }
         #object3 {
            position:fixed;
            background :yellow;
            width    :100px;
            height     :100px;
            top        :100px;
            left     :236px;
            }
      </style>
   <head>
   <body>
      <br><br><br><br><br>
      <div id='object1'>Absolute Positioning</div>
      <div id='object2'>Relative Positioning</div>
      <div id='object3'>Fixed Positioning</div>
   </body>
</html>
```





The Box Model and Layout

The box model is a nested set of properties surrounding an element. From outside to inside, you have the **margin**, then the **border**, then **padding**, then finally the object's **content**.

- **margin**.

```
margin:1px;           /* set all margins to 1 pixel */
margin-left:1px;      /* set left margin to 1 pixel */
margin-top:1px;
margin-right:1px;
margin-bottom:1px;

margin:1px 2px;       /* set top/bottom to 1px, left/right to 2px */
margin:1px 2px 3px;   /* top=1px, left/right=2px, bottom=3px */
margin:1px 2px 3px 4px; /*top=1, right=2, bottom=3, left=4 */
```

- **border**.

```
border:1px;           /* set all borders to 1 pixel */
border-left:1px;      /* set left border to 1 pixel */
border-top:1px;
border-right:1px;
border-bottom:1px;
border-width:1px;
border-color:green;
border-style:solid;   /* solid, dotted, dashed */
```

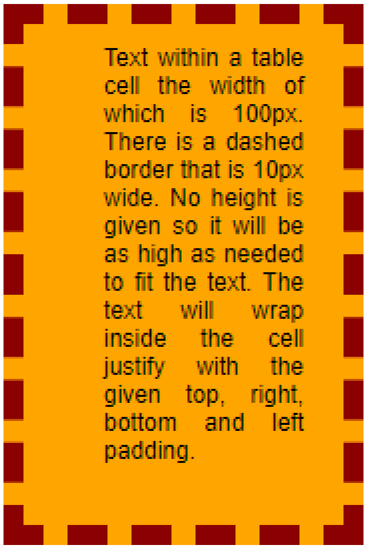
- **padding**.

```
padding:1px;          /* set all borders to 1 pixel */
padding-left:1px;     /* set left border to 1 pixel */
padding-top:1px;
padding-right:1px;
padding bottom:1px;
```

Example

```
<!DOCTYPE html>
<html>
   <head>
      <style>
```

```
        #object1 {
           display    :table-cell;
           width   :100px;
           border-style:dashed;
           border-width:10px;
           background :orange;
           border-color:darkred;
           padding    :10px 20px 30px 40px;
           font-family   :Arial;
           font-size  :12px;
           text-align :justify;
           }
     </style>
   <head>
   <body>
     <div id='object1'>Text within a table cell the width of which
is 100px.
        There is a dashed border that is 10px wide.
        No height is given so it will be as high as needed to fit
the text.
        The text will wrap inside the cell justify with the given
top, right, bottom and left padding.</div>
   </body>
</html>
```



Text within a table cell the width of which is 100px. There is a dashed border that is 10px wide. No height is given so it will be as high as needed to fit the text. The text will wrap inside the cell justify with the given top, right, bottom and left padding.