BASUDEV GODABARI DEGREE COLLEGE , KESAIBAHAL

Department of Computer Science

"SELF STUDY MODULE"

Module Details :

- Class - 5TH Semester
- Subject Name : COMPUTER SCIENCE
- Paper Name : WEB TECHNOLOGY

UNIT – 2 : STRUCTURE

2.1 Concepts of CSS

2.2 Creating style sheet, CSS properties

2.3 CSS styling (background, text format, controlling fonts),

2.4 working with the block elements and objects.

2.5 Working who lists and tables, CSS ID and class.

2.6 Box model (introduction, border properties,

2.7 padding properties, margin properties),

2.8 CSS colour, groping, Dimensions, display, positioning, floating, align, pseudo class, Navigation bar, image sprites.

Learning Objective

After Learning this unit you should be able to

1. Interactive webpage design
2. Make that content look amazing and add some cool animations and effects by learning CSS.

You Can use the Following Learning Video link related to above topic :

https://youtu.be/Edsxf_NBFrw

https://youtu.be/L1yy8QI0Aew

https://youtu.be/3Fd11f00z0s

https://youtu.be/kk_pfNxU1AM

Look related link for better understanding

You Can also use the following Books :

1. Web Technologies – Black Book – DreamTech Press
2. Matt Doyle, Beginning PHP 5.3 (wrox-Willey publishing)
3. John Duckett, Beginning HTML, XHTML, CSS and Java script.

And also you can download any book in free by using the following website.

- https://www.pdfdrive.com/

Unit-II

Cascading **S**tyle **S**heets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable.

CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs,variations in display for different devices and screen sizes as well as a variety of other effects.

CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.

Advantages of CSS

- **CSS saves time** – You can write CSS once and then reuse same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.
- **Pages load faster** – If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.
- **Easy maintenance** – To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
- **Superior styles to HTML** – CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
- **Multiple Device Compatibility** – Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
- **Global web standards** – Now HTML attributes are being deprecated and it is being recommended to use CSS. So its a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.

Who Creates and Maintains CSS?

CSS is created and maintained through a group of people within the W3C called the CSS Working Group. The CSS Working Group creates documents called specifications. When a specification has been discussed and officially ratified by the W3C members, it becomes a recommendation.

These ratified specifications are called recommendations because the W3C has no control over the actual implementation of the language. Independent companies and organizations create that software.

NOTE – The World Wide Web Consortium, or W3C is a group that makes recommendations about how the Internet works and how it should evolve.

CSS Versions

Cascading Style Sheets level 1 (CSS1) came out of W3C as a recommendation in December 1996. This version describes the CSS language as well as a simple visual formatting model for all the HTML tags.

CSS2 became a W3C recommendation in May 1998 and builds on CSS1. This version adds support for media-specific style sheets e.g. printers and aural devices, downloadable fonts, element positioning and tables.

CSS Border image property is used to add image boarder to some elements.you don't need to use any HTML code to call boarder image.A sample syntax of boarder image is as follows −

```
#borderimg {
   border: 10px solid transparent;
   padding: 15px;
}
```

The most commonly used values are shown below −

Sr.No.	Value & Description
1	**border-image-source** Used to set the image path
2	**border-image-slice** Used to slice the boarder image
3	**border-image-width** Used to set the boarder image width
4	**border-image-repeat** Used to set the boarder image as rounded, repeated and stretched

Example

Following is the example which demonstrates to set image as a border for elements.

```
<html>
   <head>
      <style>
         #borderimg1 {
            border: 10px solid transparent;
            padding: 15px;
            border-image-source: url(/css/images/border.png);
            border-image-repeat: round;
         border-image-slice: 30;
         border-image-width: 10px;
      }
      #borderimg2 {
         border: 10px solid transparent;
         padding: 15px;
         border-image-source: url(/css/images/border.png);
         border-image-repeat: round;
         border-image-slice: 30;
         border-image-width: 20px;
      }
      #borderimg3 {
         border: 10px solid transparent;
         padding: 15px;
         border-image-source: url(/css/images/border.png);
         border-image-repeat: round;
         border-image-slice: 30;
         border-image-width: 30px;
      }
    </style>
  </head>

  <body>
     <p id = "borderimg1">This is image boarder example.</p>
     <p id = "borderimg2">This is image boarder example.</p>
     <p id = "borderimg3">This is image boarder example.</p>
  </body>
</html>
```

Box sizing property is using to change the height and width of element.

Since css2, the box property has worked like as shown below –

width + padding + border = actual visible/rendered width of an element's box

height + padding + border = actual visible/rendered height of an element's box

Means when you set the height and width, it appears little bit bigger then given size cause element boarder and padding added to the element height and width.

CSS2 sizing property

```
<html>
   <head>
      <style>
         .div1 {
            width: 200px;
            height: 100px;
            border: 1px solid green;
         }
         .div2 {
            width: 200px;
            height: 100px;
            padding: 50px;
            border: 1px solid pink;
         }
      </style>
   </head>

   <body>
      <div class = "div1">TutorialsPoint.com</div><br />
      <div class = "div2">TutorialsPoint.com</div>
   </body>
</html>
```

It will produce the following result −

Above image is having same width and height of two element but giving result is different, cause second one is included padding property.

CSS3 box sizing property

```
<html>
   <head>
      <style>
         .div1 {
            width: 300px;
            height: 100px;
            border: 1px solid blue;
            box-sizing: border-box;
         }
         .div2 {
            width: 300px;
            height: 100px;
            padding: 50px;
            border: 1px solid red;
            box-sizing: border-box;
         }
      </style>
   </head>

   <body>
      <div class = "div1">TutorialsPoint.com</div><br />
      <div class = "div2">TutorialsPoint.com</div>
   </body>
</html>
```

COLOR REFERENCE

CSS3 has Supported additional color properties as follows −

- RGBA colors
- HSL colors
- HSLA colors

- Opacity

RGBA stands for **Red Green Blue Alpha**.It is an extension of CSS2,Alpha specifies the opacity of a color and parameter number is a numerical between 0.0 to 1.0. A Sample syntax of RGBA as shown below –

```
#d1 {background-color: rgba(255, 0, 0, 0.5);}
#d2 {background-color: rgba(0, 255, 0, 0.5);}
#d3 {background-color: rgba(0, 0, 255, 0.5);}
```

HSL stands for **hue, saturation, lightness**.Here Huge is a degree on the color wheel, saturation and lightness are percentage values between 0 to 100%. A Sample syntax of HSL as shown below –

```
#g1 {background-color: hsl(120, 100%, 50%);}
#g2 {background-color: hsl(120, 100%, 75%);}
#g3 {background-color: hsl(120, 100%, 25%);}
```

HSLA stands for **hue, saturation, lightness and alpha**. Alpha value specifies the opacity as shown RGBA. A Sample syntax of HSLA as shown below –

```
#g1 {background-color: hsla(120, 100%, 50%, 0.3);}
#g2 {background-color: hsla(120, 100%, 75%, 0.3);}
#g3 {background-color: hsla(120, 100%, 25%, 0.3);}
```

opacity is a thinner paints need black added to increase opacity. A sample syntax of opacity is as shown below –

```
#g1 {background-color:rgb(255,0,0);opacity:0.6;}
#g2 {background-color:rgb(0,255,0);opacity:0.6;}
#g3 {background-color:rgb(0,0,255);opacity:0.6;}
```

The following example shows rgba color propert

```
<html>
   <head>
      <style>
         #p1 {background-color:rgba(255,0,0,0.3);}
         #p2 {background-color:rgba(0,255,0,0.3);}
         #p3 {background-color:rgba(0,0,255,0.3);}
      </style>
   </head>

   <body>
      <p>RGBA colors:</p>
      <p id = "p1">Red</p>
      <p id = "p2">Green</p>
      <p id = "p3">Blue</p>
   </body>
</html>
```

This chapter teaches you how to manipulate text using CSS properties. You can set following text properties of an element –

- The **color** property is used to set the color of a text.
- The **direction** property is used to set the text direction.
- The **letter-spacing** property is used to add or subtract space between the letters that make up a word.
- The **word-spacing** property is used to add or subtract space between the words of a sentence.
- The **text-indent** property is used to indent the text of a paragraph.
- The **text-align** property is used to align the text of a document.
- The **text-decoration** property is used to underline, overline, and strikethrough text.
- The **text-transform** property is used to capitalize text or convert text to uppercase or lowercase letters.
- The **white-space** property is used to control the flow and formatting of text.
- The **text-shadow** property is used to set the text shadow around a text.

Set the Text Color

The following example demonstrates how to set the text color. Possible value could be any color name in any valid format.

```
<html>
   <head>
   </head>

   <body>
      <p style = "color:red;">
         This text will be written in red.
      </p>
   </body>
</html>
```

Set the Space between Characters

The following example demonstrates how to set the space between characters. Possible values are *normal or a number specifying space*..

```
<html>
   <head>
   </head>

   <body>
      <p style = "letter-spacing:5px;">
         This text is having space between letters.
      </p>
   </body>
</html>
```

It will produce the following result –

Set the Space between Words

The following example demonstrates how to set the space between words. Possible values are *normal or a number specifying space.*

```
<html>
   <head>
   </head>

   <body>
      <p style = "word-spacing:5px;">
         This text is having space between words.
      </p>
   </body>
</html>
```

This will produce following result –

Set the Text Indent

The following example demonstrates how to indent the first line of a paragraph. Possible values are *% or a number specifying indent space.*

```
<html>
   <head>
   </head>

   <body>
      <p style = "text-indent:1cm;">
         This text will have first line indented by 1cm and this line
will remain at
         its actual position this is done by CSS text-indent property.
      </p>
   </body>
</html>
```

It will produce the following result –

Set the Text Alignment

The following example demonstrates how to align a text. Possible values are *left, right, center, justify.*

```
<html>
   <head>
   </head>

   <body>
      <p style = "text-align:right;">
         This will be right aligned.
      </p>

      <p style = "text-align:center;">
```

```
         This will be center aligned.
      </p>

      <p style = "text-align:left;">
         This will be left aligned.
      </p>
   </body>
</html>
```

This will produce following result –

Decorating the Text

The following example demonstrates how to decorate a text. Possible values are *none, underline, overline, line-through, blin*

```
<html>
   <head>
   </head>

   <body>
      <p style = "text-decoration:underline;">
         This will be underlined
      </p>

      <p style = "text-decoration:line-through;">
         This will be striked through.
      </p>

      <p style = "text-decoration:overline;">
         This will have a over line.
      </p>

      <p style = "text-decoration:blink;">
         This text will have blinking effect
      </p>
   </body>
</html>
```

This will produce following result –

Set the Text Cases

The following example demonstrates how to set the cases for a text. Possible values are *none, capitalize, uppercase, lowercase*.

```
<html>
   <head>
   </head>

   <body>
      <p style = "text-transform:capitalize;">
         This will be capitalized
      </p>

      <p style = "text-transform:uppercase;">
         This will be in uppercase
      </p>

      <p style = "text-transform:lowercase;">
         This will be in lowercase
      </p>
   </body>
</html>
```

This will produce following result −

Set the White Space between Text

The following example demonstrates how white space inside an element is handled. Possible values are *normal, pre, nowrap*.

```
<html>
   <head>
   </head>

   <body>
      <p style = "white-space:pre;">
         This text has a line break and the white-space pre setting
         tells the browser to honor it just like the HTML pre tag.
      </p>
   </body>
</html>
```

This will produce following result −

Set the Text Shadow

The following example demonstrates how to set the shadow around a text. This may not be supported by all the browsers.

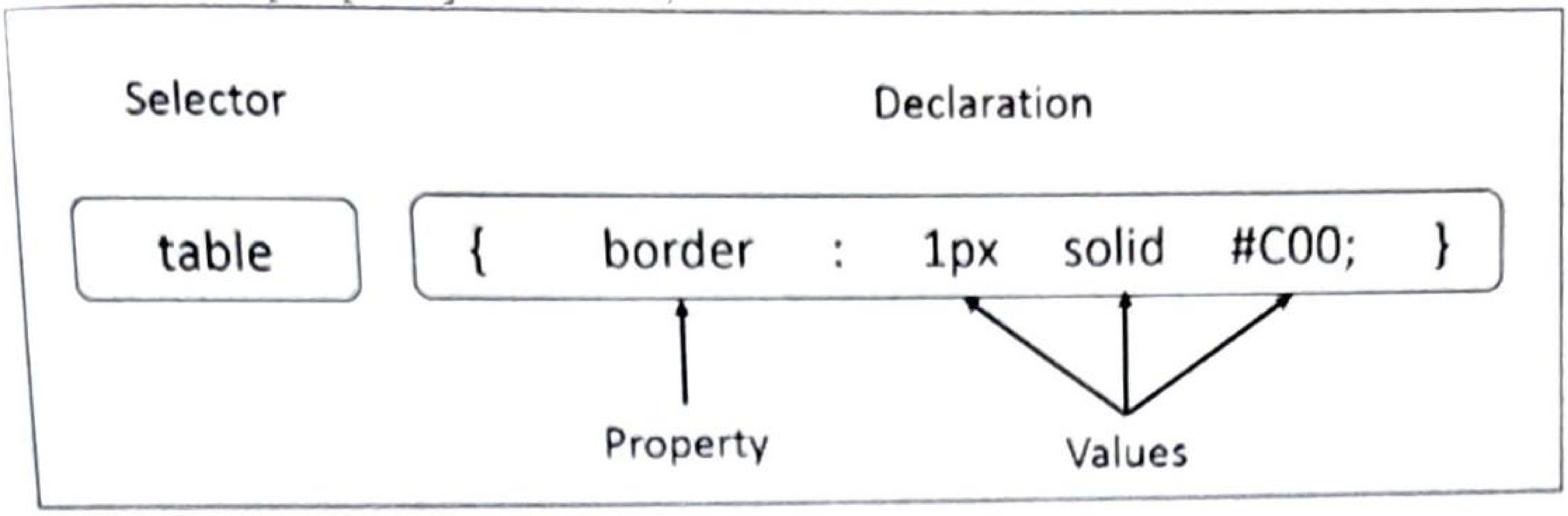
```
<html>
   <head>
   </head>

   <body>
      <p style = "text-shadow:4px 4px 8px blue;">
         If your browser supports the CSS text-shadow property,
         this text will have a  blue shadow.
      </p>
   </body>
</html>
```

A CSS comprises of style rules that are interpreted by the browser and then applied to the corresponding elements in your document. A style rule is made of three parts −

- **Selector** − A selector is an HTML tag at which a style will be applied. This could be any tag like <h1> or <table> etc.
- **Property** − A property is a type of attribute of HTML tag. Put simply, all the HTML attributes are converted into CSS properties. They could be *color*, *border* etc.
- **Value** − Values are assigned to properties. For example, *color* property can have value either *red* or *#F1F1F1* etc.

You can put CSS Style Rule Syntax as follows −

```
selector { property: value }
```



Example − You can define a table border as follows −

```
table{ border :1px solid #C00; }
```

Here table is a selector and border is a property and given value *1px solid #C00* is the value of that property.

You can define selectors in various simple ways based on your comfort. Let me put these selectors one by one.

The Type Selectors

This is the same selector we have seen above. Again, one more example to give a color to all level 1 headings −

```
h1 {
   color: #36CFFF;
}
```

The Universal Selectors

Rather than selecting elements of a specific type, the universal selector quite simply matches the name of any element type −

```
* {
   color: #000000;
}
```

This rule renders the content of every element in our document in black.

The Descendant Selectors

Suppose you want to apply a style rule to a particular element only when it lies inside a particular element. As given in the following example, style rule will apply to <em> element only when it lies inside <ul> tag.

```
ul em {
   color: #000000;
}
```

The Class Selectors

You can define style rules based on the class attribute of the elements. All the elements having that class will be formatted according to the defined rule.

```
.black {
   color: #000000;
}
```

This rule renders the content in black for every element with class attribute set to *black* in our document. You can make it a bit more particular. For example –

```
h1.black {
   color: #000000;
}
```

This rule renders the content in black for only <h1> elements with class attribute set to *black*.

You can apply more than one class selectors to given element. Consider the following example –

```
<p class = "center bold">
   This para will be styled by the classes center and bold.
</p>
```

The ID Selectors

You can define style rules based on the *id* attribute of the elements. All the elements having that *id* will be formatted according to the defined rule.

```
#black {
   color: #000000;
}
```

This rule renders the content in black for every element with *id* attribute set to *black* in our document. You can make it a bit more particular. For example –

```
h1#black {
   color: #000000;
}
```

This rule renders the content in black for only <h1> elements with *id* attribute set to *black*.

The true power of *id* selectors is when they are used as the foundation for descendant selectors, For example –

```
#black h2 {
   color: #000000;
}
```

In this example all level 2 headings will be displayed in black color when those headings will lie with in tags having *id* attribute set to *black*.

The Child Selectors

You have seen the descendant selectors. There is one more type of selector, which is very similar to descendants but have different functionality. Consider the following example –

```
body > p {
   color: #000000;
}
```

This rule will render all the paragraphs in black if they are direct child of <body> element. Other paragraphs put inside other elements like <div> or <td> would not have any effect of this rule.

The Attribute Selectors

You can also apply styles to HTML elements with particular attributes. The style rule below will match all the input elements having a type attribute with a value of *text* –

```
input[type = "text"] {
   color: #000000;
}
```

The advantage to this method is that the <input type = "submit" /> element is unaffected, and the color applied only to the desired text fields.

There are following rules applied to attribute selector.

- **p[lang]** – Selects all paragraph elements with a *lang* attribute.
- **p[lang="fr"]** – Selects all paragraph elements whose *lang* attribute has a value of exactly "fr".
- **p[lang~="fr"]** – Selects all paragraph elements whose *lang* attribute contains the word "fr".
- **p[lang|="en"]** – Selects all paragraph elements whose *lang* attribute contains values that are exactly "en", or begin with "en-".

Multiple Style Rules

You may need to define multiple style rules for a single element. You can define these rules to combine multiple properties and corresponding values into a single block as defined in the following example –

```
h1 {
   color: #36C;
   font-weight: normal;
   letter-spacing: .4em;
   margin-bottom: 1em;
   text-transform: lowercase;
}
```

Here all the property and value pairs are separated by a **semicolon (;)**. You can keep them in a single line or multiple lines. For better readability, we keep them in separate lines.

For a while, don't bother about the properties mentioned in the above block. These properties will be explained in the coming chapters and you can find complete detail about properties in CSS References

Grouping Selectors

You can apply a style to many selectors if you like. Just separate the selectors with a comma, as given in the following example –

```
h1, h2, h3 {
   color: #36C;
   font-weight: normal;
   letter-spacing: .4em;
   margin-bottom: 1em;
   text-transform: lowercase;
}
```

This define style rule will be applicable to h1, h2 and h3 element as well. The order of the list is irrelevant. All the elements in the selector will have the corresponding declarations applied to them.

You can combine the various *id* selectors together as shown below –

```
#content, #footer, #supplement {
   position: absolute;
   left: 510px;
   width: 200px;
}
```

CSS3 supported to add shadow to text or elements.Shadow property has divided as follows –

- Text shadow
- Box Shadow

Text shadow

CSS3 supported to add shadow effects to text. Following is the example to add shadow effects to text

```
<html>
   <head>
      <style>
         h1 {
            text-shadow: 2px 2px;

         }
         h2 {
            text-shadow: 2px 2px red;

         }
         h3 {
            text-shadow: 2px 2px 5px red;

         }
         h4 {
            color: white;
            text-shadow: 2px 2px 4px #000000;

         }
         h5 {
            text-shadow: 0 0 3px #FF0000;

         }
         h6 {
            text-shadow: 0 0 3px #FF0000, 0 0 5px #0000FF;

         }
         p {
            color: white;
            text-shadow: 1px 1px 2px black, 0 0 25px blue, 0 0 5px
darkblue;
         }
      </style>
   </head>
   <body>
      <h1>Tutorialspoint.com</h1>
      <h2>Tutorialspoint.com</h2>
      <h3>Tutorialspoint.com</h3>
      <h4>Tutorialspoint.com</h4>
      <h5>Tutorialspoint.com</h5>
      <h6>Tutorialspoint.com</h6>
      <p>Tutorialspoint.com</p>
   </body>
</html>
```

It will produce the following result –

box shadow

Used to add shadow effects to elements, Following is the example to add shadow effects to elem

```
<html>
   <head>
      <style>
         div {
            width: 300px;
            height: 100px;
            padding: 15px;
            background-color: red;
            box-shadow: 10px 10px;
         }
      </style>
   </head>

   <body>
      <div>This is a div element with a box-shadow</div>
   </body>
</html>
```

This tutorial will teach you how to set different properties of an HTML table using CSS. You can set following properties of a table –

- The **border-collapse** specifies whether the browser should control the appearance of the adjacent borders that touch each other or whether each cell should maintain its style.
- The **border-spacing** specifies the width that should appear between table cells.
- The **caption-side** captions are presented in the <caption> element. By default, these are rendered above the table in the document. You use the *caption-side* property to control the placement of the table caption.
- The **empty-cells** specifies whether the border should be shown if a cell is empty.
- The **table-layout** allows browsers to speed up layout of a table by using the first width properties it comes across for the rest of a column rather than having to load the whole table before rendering it.