HTML 5 Handbook

Raman Deep Singh

You can write HTML 5 Code in Notepad and save file as .html and run it in any browser like Edge, Firefox, Safari or Chrome.

Basic Structure of HTML 5 Webpage

```
<!DOCTYPE html>
<html lang="en">
<head>
   <title>A simple HTML document</title>
</head>
<body>
   <p>Hello World!<p>
</body>
</html>
```

Technically, an HTML element is the collection of start tag, its attributes, an end tag and everything in between. On the other hand an HTML tag (either opening or closing) is used to mark the start or end of an element, as you can see in the above illustration.

```
<!DOCTYPE html>
<html lang="en">
<head>
   <title>A simple HTML document</title>
</head>
<body>
<p>This is a paragraph.</p>
<P>This is also a valid paragraph.</P>
</body>
</html>
```

Empty HTML Elements

```
<p>This paragraph contains <br> a line break.</p>
<img src="images/sky.jpg" alt="Cloudy Sky">
<input type="text" name="username">
```

Nested HTML Elements

```
<p>Here is some <b>bold</b> text.</p>

<p>Here is some <em>emphasized</em> text.</p>

<p>Here is some <mark>highlighted</mark> text.</p>
```

```
<p><strong>These tags are nested properly.</strong></p>

<p><strong>These tags are not nested properly.</p></strong>
```

```
<!-- This is an HTML comment -->

<!-- This is a multi-line HTML comment

    that spans across more than one line -->

<p>This is a normal piece of text.</p>
```

Attributes define additional characteristics or properties of the element such as width and height of an image.

```
<img src="images/smiley.png" width="30" height="30" alt="Smiley">

<a href="https://www.google.com/" title="Search Engine">Google</a>

<abbr title="Hyper Text Markup Language">HTML</abbr>

<input type="text" value="John Doe">
```

Examples of some commonly used Boolean attributes are checked, disabled, readonly, required, etc.

```
<input type="email" required>

<input type="submit" value="Submit" disabled>

<input type="checkbox" checked>

<input type="text" value="Read only text" readonly>
```

The id attribute is used to give a unique name or identifier to an element within a document.

```
<input type="text" id="firstName">

<div id="container">Some content</div>

<p id="infoText">This is a paragraph.</p>
```

Like id attribute, the class attribute is also used to identify elements. But unlike id, the class attribute does not have to be unique in the document. This means you can apply the same class to multiple elements in a document, as shown in the following example:

```
<input type="text" class="highlight">

<div class="box highlight">Some content</div>

<p class="highlight">This is a paragraph.</p>
```

The title attribute to is used to provide advisory text about an element or its content.

```
<abbr title="World Wide Web Consortium">W3C</abbr>

<a href="images/kites.jpg" title="Click to view a larger image">
   <img src="images/kites-thumb.jpg" alt="kites">
</a>
```

The style attribute allows you to specify CSS styling rules such as color, font, border, etc. directly within the element.

```
<p style="color: blue;">This is a paragraph.</p>

<img src="images/sky.jpg" style="width: 300px;" alt="Cloudy Sky">

<div style="border: 1px solid red;">Some content</div>
```

HTML offers six levels of heading tags, <h1> through <h6>; the lower the heading level number, the greater its importance — therefore <h1> tag defines the most important heading, whereas the <h6> tag defines the least important heading in the document.

```
<h1>Heading level 1</h1>

<h2>Heading level 2</h2>

<h3>Heading level 3</h3>

<h4>Heading level 4</h4>

<h5>Heading level 5</h5>

<h6>Heading level 6</h6>
```

Paragraph element is used to publish text on the web pages.

Paragraphs are defined with the <p> tag. Paragraph tag is a very basic and typically the first tag you will need to publish your text on the web pages.

```
<p>This is a paragraph.</p>

<p>This is another paragraph.</p>
```

The <br> tag is used to insert a line break on the web page.

Since the <br> is an empty element, so there is no need of corresponding </br> tag

```
<p>This is a paragraph <br> with line break.</p>

<p>This is <br>another paragraph <br> with line breaks.</p>
```

You can use the <hr> tag to create horizontal rules or lines to visually separate content sections on a web page. Like <br>, the <hr> tag is also an empty element.

```
<p>This is a paragraph.</p>

<hr>
```

This is another paragraph.

---

insert whitespaces

Insert   for creating extra consecutive spaces, while insert <br> tag for creating line breaks on your web pages

This paragraph has multiple   spaces.

This paragraph has multiple

line

breaks.

---

you can use the <pre> tag to display spaces, tabs, line breaks, etc. exactly as written in the HTML file. It is very helpful in presenting text where spaces and line breaks are important like poem or code.

<pre>

  Twinkle, twinkle, little star,

  How I wonder what you are!

  Up above the world so high,

  Like a diamond in the sky.

</pre>

A link or hyperlink is a connection from one web resource to another. Links allow users to move seamlessly from one page to another, on any server anywhere in the world.

A link has two ends, called anchors. The link starts at the source anchor and points to the destination anchor, which may be any web resource, for example, an image, an audio or video clip, a PDF file, an HTML document or an element within the document itself, and so on.

Google Search

Cpp Courses

  <img src="kites-thumb.jpg" alt="kites">

The target attribute tells the browser where to open the linked document. There are four defined targets, and each target name starts with an underscore(_) character:

_blank — Opens the linked document in a new window or tab.

_parent — Opens the linked document in the parent window.

_self — Opens the linked document in the same window or tab as the source document. This is the default, hence it is not necessary to explicitly specify this value.

_top — Opens the linked document in the full browser window.

```
<a href="/about-us.php" target="_top">About Us</a>
<a href="https://www.google.com/" target="_blank">Google</a>
<a href="images/sky.jpg" target="_parent">
   <img src="sky-thumb.jpg" alt="Cloudy Sky">
</a>
```

You can also create bookmark anchors to allow users to jump to a specific section of a web page. Bookmarks are especially helpful if you have a very long web page.

```
<a href="#sectionA">Jump to Section A</a>
<h2 id="sectionA">Section A</h2>
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit...</p>
```

You can also create the file download link in exactly the same fashion as placing text links. Just point the destination URL to the file you want to be available for download.

```
<a href="downloads/test.zip">Download Zip file</a>
<a href="downloads/masters.pdf">Download PDF file</a>
<a href="downloads/sample.jpg">Download Image file</a>
```

HTML provides several tags that you can use to make some text on your web pages to appear differently than normal text, for example, you can use the tag <b> to make the text bold, tag <i> to make the text italic, tag <mark> to highlight the text, tag <code> to display a fragment of computer code, tags <ins> and <del> for marking editorial insertions and deletions, and more.

```
<p>This is <b>bold text</b>.</p>
<p>This is <strong>strongly important text</strong>.</p>
<p>This is <i>italic text</i>.</p>
<p>This is <em>emphasized text</em>.</p>
<p>This is <mark>highlighted text</mark>.</p>
<p>This is <code>computer code</code>.</p>
<p>This is <small>smaller text</small>.</p>
<p>This is <sub>subscript</sub> and <sup>superscript</sup> text.</p>
<p>This is <del>deleted text</del>.</p>
<p>This is <ins>inserted text</ins>.</p>
```

---

difference between strong and b tag
Both and <b> tags render the enclosed text in a bold typeface by default, but the

tag indicates that its contents have strong importance, whereas the tag is simply used to draw the reader's attention without conveying any special importance.

WARNING! Please proceed with caution.

The concert will be held at Hyde Park in London.

---

difference between em and i tag

both and tags render the enclosed text in italic type by default, but the tag indicates that its contents have stressed emphasis compared to surrounding text, whereas the tag is used for marking up text that is set off from the normal text for readability reasons, such as a technical term, an idiomatic phrase from another language, a thought, etc.

Cats are cute animals.

The Royal Cruise sailed last night.

You can easily format the quotation blocks from other sources with the HTML <blockquote> tag.

Blockquotes are generally displayed with indented left and right margins, along with a little extra space added above and below.

<blockquote>

  Learn from yesterday, live for today, hope for tomorrow. The important thing is not to stop questioning.

  <cite>— Albert Einstein</cite>

</blockquote>

An abbreviation is a shortened form of a word, phrase, or name.

You can use the <abbr> tag to denote an abbreviation. The title attribute is used inside this tag to provide the full expansion of the abbreviation, which is displayed by the browsers as a tooltip when the mouse cursor is hovered over the element.

The <abbr title="World Wide Web Consortium">W3C</abbr> is the main international standards organization for the <abbr title="World Wide Web">WWW or W3</abbr>. It was was founded by Tim Berners-Lee.

Web pages often include street or postal addresses. HTML provides a special tag <address> to represent contact information (physical and/or digital) for a person, people or organization.

This tag should ideally used to display contact information related to the document itself, such as article's author. Most browsers display an address block in italic.

<address>

Mozilla Foundation

331 E. Evelyn Avenue

Mountain View, CA 94041, USA

</address>

Style information can either be attached as a separate document or embedded in the HTML document itself. These are the three methods of implementing styling information to an HTML document.

Inline styles — Using the style attribute in the HTML start tag.

Embedded style — Using the <style> element in the head section of the document.

External style sheet — Using the <link> element, pointing to an external CSS files.

Inline styles are used to apply the unique style rules to an element, by putting the CSS rules directly into the start tag. It can be attached to an element using the style attribute.

The style attribute includes a series of CSS property and value pairs. Each property: value pair is separated by a semicolon (;), just as you would write into an embedded or external style sheet. But it needs to be all in one line i.e. no line break after the semicolon.

```
<h1 style="color:red; font-size:30px;">This is a heading</h1>
<p style="color:green; font-size:18px;">This is a paragraph.</p>
<div style="color:green; font-size:18px;">This is some text.</div>
```

Embedded or internal style sheets only affect the document they are embedded in.

Embedded style sheets are defined in the <head> section of an HTML document using the <style> tag. You can define any number of <style> elements inside the <head> section.

```
<head>
  <style>
    body { background-color: YellowGreen; }
                h1 { color: blue; }
    p { color: red; }
  </style>
</head>
```

An external style sheet is ideal when the style is applied to many pages.

An external style sheet holds all the style rules in a separate document that you can link from any HTML document on your site. External style sheets are the most flexible because with an external style sheet, you can change the look of an entire website by updating just one file.

You can attach external style sheets in two ways — linking and importing:

Linking External Style Sheets

An external style sheet can be linked to an HTML document using the <link> tag.

The <link> tag goes inside the <head> section, as shown here:

```
<head>
  <link rel="stylesheet" href="css/style.css">
</head>
```

The @import rule is another way of loading an external style sheet. The @import statement instructs the browser to load an external style sheet and use its styles.

You can use it in two ways. The simplest way is to use it within the <style> element in your <head> section. Note that, other CSS rules may still be included in the <style> element.

```
<style>
  @import url("css/style.css");
  p {
    color: blue;
    font-size: 16px;
  }
</style>
```

Similarly, you can use the @import rule to import a style sheet within another style sheet.

```
@import url("css/layout.css");
@import url("css/color.css");
body {
   color: blue;
   font-size: 14px;
}
```

The <img> tag is used to insert images in the HTML documents. It is an empty element and contains attributes only.

```
<img src="kites.jpg" alt="Flying Kites">
<img src="sky.jpg" alt="Cloudy Sky">
<img src="balloons.jpg" alt="Balloons">
```

The width and height attributes are used to specify the width and height of an image.

The values of these attributes are interpreted in pixels by default.

```
<img src="kites.jpg" alt="Flying Kites" width="300" height="300">
<img src="sky.jpg" alt="Cloudy Sky" width="250" height="150">
<img src="balloons.jpg" alt="Balloons" width="200" height="200">
```

An image map allows you to define hotspots on an image that acts just like a hyperlink.

The basic idea behind creating image map is to provide an easy way of linking various parts of an image without dividing it into separate image files. For example, a map of the world may have each country hyperlinked to further information about that country.

```
<img src="objects.png" usemap="#objects" alt="Objects">
<map name="objects">
   <area shape="circle" coords="137,231,71" href="clock.html" alt="Clock">
   <area shape="poly" coords="363,146,273,302,452,300" href="sign.html" alt="Sign">
   <area shape="rect" coords="520,160,641,302" href="book.html" alt="Book">
</map>
```

HTML table allows you to arrange data into rows and columns. They are commonly used to display tabular data like product listings, customer's details, financial reports, and so on.

You can create a table using the <table> element. Inside the <table> element, you can use the <tr> elements to create rows, and to create columns inside a row you can use the <td> elements. You can also define a cell as a header for a group of table cells using the <th> element.

```
<table>
  <tr>
    <th>No.</th>
    <th>Name</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>1</td>
    <td>Peter Parker</td>
    <td>16</td>
  </tr>
  <tr>
    <td>2</td>
    <td>Clark Kent</td>
    <td>34</td>
  </tr>
</table>
```

Spanning allow you to extend table rows and columns across multiple other rows and columns.

Normally, a table cell cannot pass over into the space below or above another table cell. But, you can use the rowspan or colspan attributes to span multiple rows or columns in a table.

```
<table>
  <tr>
    <th>Name</th>
```

```
    <th colspan="2">Phone</th>
  </tr>
  <tr>
    <td>John Carter</td>
    <td>5550192</td>
    <td>5550152</td>
  </tr>
</table>
```

Similarly, you can use the rowspan attribute to create a cell that spans more than one row. Let's try out an example to understand how row spanning basically works:

```
<table>
  <tr>
    <th>Name:</th>
    <td>John Carter</td>
  </tr>
  <tr>
    <th rowspan="2">Phone:</th>
    <td>55577854</td>
  </tr>
  <tr>
    <td>55577855</td>
  </tr>
</table>
```

You can specify a caption (or title) for your tables using the <caption> element.

The <caption> element must be placed directly after the opening <table> tag. By default, caption appears at the top of the table, but you can change its position using the CSS caption-side property.

```
<table>
  <caption>Users Info</caption>
```

```
  <tr>
    <th>No.</th>
    <th>Name</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>1</td>
    <td>Peter Parker</td>
    <td>16</td>
  </tr>
  <tr>
    <td>2</td>
    <td>Clark Kent</td>
    <td>34</td>
  </tr>
</table>
```

HTML provides a series of tags <thead>, <tbody>, and <tfoot> that helps you to create more structured table, by defining header, body and footer regions, respectively.

```
<table>
  <thead>
    <tr>
      <th>Items</th>
      <th>Expenditure</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Stationary</td>
      <td>2,000</td>
    </tr>
```

```
      <tr>
        <td>Furniture</td>
        <td>10,000</td>
      </tr>
    </tbody>
    <tfoot>
      <tr>
        <th>Total</th>
        <td>12,000</td>
      </tr>
    </tfoot>
</table>
```

HTML lists are used to present list of information in well formed and semantic way. There are three different types of list in HTML and each one has a specific purpose and meaning.

Unordered list — Used to create a list of related items, in no particular order.

Ordered list — Used to create a list of related items, in a specific order.

Description list — Used to create a list of terms and their descriptions.

An unordered list created using the <ul> element, and each list item starts with the <li> element.

The list items in unordered lists are marked with bullets.

```
<ul>
  <li>Chocolate Cake</li>
  <li>Black Forest Cake</li>
  <li>Pineapple Cake</li>
</ul>
```

You can also change the bullet type in your unordered list using the CSS list-style-type property. The following style rule changes the type of bullet from the default disc to square:

```
ul {
   list-style-type: square;
}
```

An ordered list created using the <ol> element, and each list item starts with the <li> element. Ordered lists are used when the order of the list's items is important.

The list items in an ordered list are marked with numbers. Here's an example:

```
<ol>
   <li>Fasten your seatbelt</li>
   <li>Starts the car's engine</li>
   <li>Look around and go</li>
</ol>
```

The numbering of items in an ordered list typically starts with 1. However, if you want to change that you can use the start attribute, as shown in the following example:

```
<ol start="10">
   <li>Mix ingredients</li>
   <li>Bake in oven for an hour</li>
   <li>Allow to stand for ten minutes</li>
</ol>
```

A description list is a list of items with a description or definition of each item.

The description list is created using <dl> element. The <dl> element is used in conjunction with the <dt> element which specify a term, and the <dd> element which specify the term's definition.

```
<dl>
   <dt>Bread</dt>
   <dd>A baked food made of flour.</dd>
```

```
  <dt>Coffee</dt>
  <dd>A drink made from roasted coffee beans.</dd>
</dl>
```

HTML Forms are required to collect different kinds of user inputs, such as contact details like name, email address, phone numbers, or details like credit card information, etc.

Forms contain special elements called controls like inputbox, checkboxes, radio-buttons, submit buttons, etc. Users generally complete a form by modifying its controls e.g. entering text, selecting items, etc. and submitting this form to a web server for further processing.

The <form> tag is used to create an HTML form.

```
<form>
  <label>Username: <input type="text"></label>
  <label>Password: <input type="password"></label>
  <input type="submit" value="Submit">
</form>
```

Input Element

This is the most commonly used element within HTML forms.

It allows you to specify various types of user input fields, depending on the type attribute. An input element can be of type text field, password field, checkbox, radio button, submit button, reset button, file select box, as well as several new input types introduced in HTML5.

Text fields are one line areas that allow the user to input text.

Single-line text input controls are created using an <input> element, whose type attribute has a value of text.

```
<form>
  <label for="username">Username:</label>
  <input type="text" name="username" id="username">
```

```
</form>
```

Password fields are similar to text fields. The only difference is; characters in a password field are masked, i.e. they are shown as asterisks or dots. This is to prevent someone else from reading the password on the screen. This is also a single-line text input controls created using an <input> element whose type attribute has a value of password.

```
<form>
  <label for="user-pwd">Password:</label>
  <input type="password" name="user-password" id="user-pwd">
</form>
```

Radio buttons are used to let the user select exactly one option from a pre-defined set of options. It is created using an <input> element whose type attribute has a value of radio.

```
<form>
  <input type="radio" name="gender" id="male">
  <label for="male">Male</label>
  <input type="radio" name="gender" id="female">
  <label for="female">Female</label>
</form>
```

Checkboxes allows the user to select one or more option from a pre-defined set of options. It is created using an <input> element whose type attribute has a value of checkbox.

```
<form>
  <input type="checkbox" name="sports" id="soccer">
  <label for="soccer">Soccer</label>
  <input type="checkbox" name="sports" id="cricket">
  <label for="cricket">Cricket</label>
  <input type="checkbox" name="sports" id="baseball">
  <label for="baseball">Baseball</label>
</form>
```

The file fields allow a user to browse for a local file and send it as an attachment with the form data. Web browsers such as Google Chrome and Firefox render a file select input field with a Browse button that enables the user to navigate the local hard drive and select a file.

This is also created using an <input> element, whose type attribute value is set to file.

```
<form>
   <label for="file-select">Upload:</label>
   <input type="file" name="upload" id="file-select">
</form>
```

Textarea is a multiple-line text input control that allows a user to enter more than one line of text. Multi-line text input controls are created using an <textarea> element.

```
<form>
   <label for="address">Address:</label>
   <textarea rows="3" cols="30" name="address" id="address"></textarea>
</form>
```

A select box is a dropdown list of options that allows user to select one or more option from a pull-down list of options. Select box is created using the <select> element and <option> element.

The <option> elements within the <select> element define each list item.

```
<form>
   <label for="city">City:</label>
   <select name="city" id="city">
      <option value="sydney">Sydney</option>
      <option value="melbourne">Melbourne</option>
      <option value="cromwell">Cromwell</option>
   </select>
```

```
</form>
```

A submit button is used to send the form data to a web server. When submit button is clicked the form data is sent to the file specified in the form's action attribute to process the submitted data.

A reset button resets all the forms control to default values.

```
<form action="action.php" method="post">
  <label for="first-name">First Name:</label>
  <input type="text" name="first-name" id="first-name">
  <input type="submit" value="Submit">
  <input type="reset" value="Reset">
</form>
```

Grouping Form Controls

You also group logically related controls and labels within a web form using the <legend> element. Grouping form controls into categories makes it easier for users to locate a control which makes the form more user-friendly.

```
<form>
  <fieldset>
    <legend>Contact Details</legend>
    <label>Email Address: <input type="email" name="email"></label>
    <label>Phone Number: <input type="text" name="phone"></label>
  </fieldset>
</form>
```

An iframe or inline frame is used to display external objects including other web pages within a web page. An iframe pretty much acts like a mini web browser within a web browser. Also, the content inside an iframe exists entirely independent from the surrounding elements.

```
<iframe src="hello.html"></iframe>
```

The height and width attributes are used to specify the height and width of the iframe.

```
<iframe src="hello.html" width="400" height="200"></iframe>
```

The iframe has a border around it by default. However, if you want to modify or remove the iframe borders, the best way is to use the CSS border property.

```
<iframe src="hello.html" style="border: none;"></iframe>
```

Similarly, you can use the border property to add the borders of your choice to an iframe. The following example will render the iframe with 2 pixels blue border.

```
<iframe src="hello.html" style="border: 2px solid blue;"></iframe>
```

An iframe can also be used as a target for the hyperlinks.

An iframe can be named using the name attribute. This implies that when a link with a target attribute with that name as value is clicked, the linked resource will open in that iframe.

```
<iframe src="demo-page.html" name="myFrame"></iframe>

<p><a href="https://www.tutorialrepublic.com" target="myFrame">Open TutorialRepublic.com</a></p>
```

A Document Type Declaration, or DOCTYPE for short, is an instruction to the web browser about the version of markup language in which a web page is written.

A DOCTYPE declaration appears at the top of a web page before all other elements. According to the HTML specification or standards, every HTML document requires a valid document type declaration to insure that your web pages are displayed the way they are intended to be displayed.

The doctype declaration is usually the very first thing defined in an HTML document (even before the opening <html> tag); however the doctype declaration itself is not an HTML tag.

The DOCTYPE for HTML5 is very short, concise, and case-insensitive.

```
<!DOCTYPE html>
```

```
<html lang="en">
<head>
   <meta charset="utf-8">
   <title><!-- Insert your title here --></title>
</head>
<body>
   <!-- Insert your content here -->
</body>
</html>
```

Creating Website Layouts

Creating a website layout is the activity of positioning the various elements that make a web page in a well-structured manner and give appealing look to the website.

You have seen most websites on the internet usually display their content in multiple rows and columns, formatted like a magazine or newspaper to provide the users a better reading and writing environment. This can be easily achieved by using the HTML tags, such as <table>, , <header>, <footer>, <section>, etc. and adding some CSS styles to them.

HTML Table Based Layout

Table provides the simplest way for creating layouts in HTML. Generally, this involves the process of putting the contents such as text, images, and so on into rows and columns.

The following layout is created using an HTML table with 3 rows and 2 columns — the first and last row spans both columns using the table's colspan attribute:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>HTML Table Layout</title>
</head>
<body style="margin:0px;">
   <table style="width:100%; border-collapse:collapse; font:14px Arial,sans-serif;">
```

```
    <tr>
      <td colspan="2" style="padding:10px 20px; background-color:#acb3b9;">
        <h1 style="font-size:24px;">Programming Languages Tutorials</h1>
      </td>
    </tr>
    <tr style="height:170px;">
      <td style="width:20%; padding:20px; background-color:#d4d7dc; vertical-align: top;">
        <ul style="list-style:none; padding:0px; line-height:24px;">
          <li><a href="#" style="color:#333;">Home</a></li>
          <li><a href="#" style="color:#333;">About</a></li>
          <li><a href="#" style="color:#333;">Contact</a></li>
        </ul>
      </td>
      <td style="padding:20px; background-color:#f2f2f2; vertical-align:top;">
        <h2>Welcome to our site</h2>
        <p>Here you will learn how to create websites...</p>
      </td>
    </tr>
    <tr>
      <td colspan="2" style="padding:5px; background-color:#acb3b9; text-align:center;">
        <p>copyright &copy; cppcourses.com</p>
      </td>
    </tr>
  </table>
</body>
</html>
```

HTML Div Based Layout

Using the elements is the most common method of creating layouts in HTML. The (stands for division) element is used for marking out a block of content, or set of other elements inside an HTML document. It can contain further other div elements if required.

The following example uses the div elements to create a multiple column layout.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>HTML Div Layout</title>
<style>
   body {
      font: 14px Arial,sans-serif;
      margin: 0px;
   }
   .header {
      padding: 10px 20px;
      background: #acb3b9;
   }
   .header h1 {
      font-size: 24px;
   }
   .container {
      width: 100%;
      background: #f2f2f2;
   }
   .nav, .section {
      float: left;
      padding: 20px;
      min-height: 170px;
      box-sizing: border-box;
   }
   .nav {
      width: 20%;
      background: #d4d7dc;
```

```
    }
    .section {
        width: 80%;
    }
    .nav ul {
        list-style: none;
        line-height: 24px;
        padding: 0px;
    }
    .nav ul li a {
        color: #333;
    }
    .clearfix:after {
        content: ".";
        display: block;
        height: 0;
        clear: both;
        visibility: hidden;
    }
    .footer {
        background: #acb3b9;
        text-align: center;
        padding: 5px;
    }
</style>
</head>
<body>
    <div class="container">
        <div class="header">
            <h1>Programming Languages Tutorials</h1>
        </div>
```

```
    <div class="wrapper clearfix">
      <div class="nav">
        <ul>
          <li><a href="#">Home</a></li>
          <li><a href="#">About</a></li>
          <li><a href="#">Contact</a></li>
        </ul>
      </div>
      <div class="section">
        <h2>Welcome to our site</h2>
        <p>Here you will learn how to create websites...</p>
      </div>
    </div>
    <div class="footer">
      <p>copyright &copy; cppcourses.com</p>
    </div>
  </div>
</body>
</html>
```

Using the HTML5 Structural Elements

HTML5 has introduced the new structural elements such as <header>, <footer>, <nav>, <section>, etc. to define the different parts of a web page in a more semantic way.

You can consider these elements as a replacement for commonly used classes such as .header, .footer, .nav, .section, etc.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>HTML5 Web Page Layout</title>
<style>
```

```
body {
    font: 14px Arial,sans-serif;
    margin: 0px;
}
header {
    padding: 10px 20px;
    background: #acb3b9;
}
header h1 {
    font-size: 24px;
}
.container {
    width: 100%;
    background: #f2f2f2;
}
nav, section {
    float: left;
    padding: 20px;
    min-height: 170px;
    box-sizing: border-box;
}
section {
    width: 80%;
}
nav {
    width: 20%;
    background: #d4d7dc;
}
nav ul {
    list-style: none;
    line-height: 24px;
```

```
        padding: 0px;
    }
    nav ul li a {
        color: #333;
    }
    .clearfix:after {
        content: ".";
        display: block;
        height: 0;
        clear: both;
        visibility: hidden;
    }
    footer {
        background: #acb3b9;
        text-align: center;
        padding: 5px;
    }
</style>
</head>
<body>
    <div class="container">
        <header>
            <h1>Programming Languages Tutorials</h1>
        </header>
        <div class="wrapper clearfix">
            <nav>
                <ul>
                    <li><a href="#">Home</a></li>
                    <li><a href="#">About</a></li>
                    <li><a href="#">Contact</a></li>
                </ul>
```

```
        </nav>
        <section>
          <h2>Welcome to our site</h2>
          <p>Here you will learn how to create websites...</p>
        </section>
      </div>
      <footer>
        <p>copyright &copy; cppcourses.com</p>
      </footer>
    </div>
</body>
</html>
```

The HTML head Element

The <head> element primarily is the container for all the head elements, which provide extra information about the document (metadata), or reference to other resources that are required for the document to display or behave correctly in a web browser.

The head elements collectively describes the properties of the document such as title, provide meta information like character set, instruct the browser where to find the style sheets or scripts that allows you to extend the HTML document in a highly active and interactive ways.

The HTML elements that can be used inside the <head> element are: <title>, <base>, <link>, <style>, <meta>, <script> and the <noscript> element.

The HTML title Element

The <title> element defines the title of the document.

The title element is required in all HTML/XHTML documents to produce a valid document. Only one title element is permitted in a document and it must be placed within the <head> element. The title element contains plain text and entities; it may not contain other markup tags.

The title of the document may be used for different purposes. For example:

To display a title in the browser title bar and in the task bar.

To provide a title for the page when it is added to favorites or bookmarked.

To displays a title for the page in search-engine results.

The following example demonstrates how to place title in an HTML document.

```
<!DOCTYPE html>
<html lang="en">
<head>
   <title>A simple HTML document</title>
</head>
<body>
   <p>Hello World!</p>
</body>
</html>
```

The HTML base Element

The HTML <base> element is used to define a base URL for all relative links contained in the document, e.g. you can set the base URL once at the top of your page, and then all subsequent relative links will use that URL as a starting point. Here's an example:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <title>Defining a base URL</title>
   <base href="https://www.tutorialrepublic.com/">
</head>
<body>
   <p><a href="html-tutorial/html-head.php">HTML Head</a>.</p>
</body>
</html>
```

The HTML link Element

The <link> element defines the relationship between the current document and an external documents or resources. A common use of link element is to link to external style sheets.

```
<head>
   <title>Linking Style Sheets</title>
   <link rel="stylesheet" href="style.css">
</head>
```

The HTML style Element

The <style> element is used to define embedded style information for an HTML document. The style rules inside the <style> element specify how HTML elements render in a browser.

```
<head>
   <title>Embedding Style Sheets</title>
   <style>
      body { background-color: YellowGreen; }
      h1 { color: red; }
      p { color: green; }
   </style>
</head>
```

The HTML meta Element

The <meta> element provides metadata about the HTML document. Metadata is a set of data that describes and gives information about other data.

```
<head>
   <title>Specifying Metadata</title>
   <meta charset="utf-8">
   <meta name="author" content="John Smith">
</head>
```

The HTML script Element

The <script> element is used to define client-side script, such as JavaScript in HTML documents.

```
<head>
   <title>Adding JavaScript</title>
```

```
  <script>
    document.write("<h1>Hello World!</h1>")
  </script>
</head>
```

Defining Metadata

The <meta> tags are typically used to provide structured metadata such as a document's keywords, description, author name, character encoding, and other metadata. Any number of meta tags can be placed inside the head section of an HTML or XHTML document.

Metadata will not be displayed on the web page, but will be machine parsable, and can be used by the browsers, search engines like Google or other web services.

Meta tag typically used to declare character encoding inside HTML document.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Declaring Character Encoding</title>
  <meta charset="utf-8">
</head>
<body>
  <h1>Hello World!</h1>
</body>
</html>
```

You can also use the meta tag to clearly define who is the author or creator of the web page.

```
<head>
  <title>Defining Document's Author</title>
  <meta name="author" content="Alexander Howick">
</head>
```

Keywords and Description for Search Engines

Some search engines use metadata especially keywords and descriptions to index web pages; however this may not necessarily be true. Keywords giving extra weight to a document's keywords and description provide a short synopsis of the page.

```
<head>
   <title>Defining Keywords and Description</title>
   <meta name="keywords" content="HTML, CSS, javaScript">
   <meta name="description" content="Easy to understand tutorials and references on HTML, CSS, javaScript and more...">
</head>
```

To embed JavaScript in an HTML file, just add the code as the content of a <script> element.

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="utf-8">
   <title>Embedding JavaScript</title>
</head>
<body>
   <div id="greet"></div>
   <script>
      document.getElementById("greet").innerHTML = "Hello World!";
   </script>
</body>
</html>
```

Calling External JavaScript File

You can also place your JavaScript code into a separate file (with a .js extension), and call that file in your HTML document through the src attribute of the <script> tag.

```
<!DOCTYPE html>
```

```
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Linking External JavaScript</title>
</head>
<body>
  <div id="greet"></div>
  <script src="hello.js"></script>
</body>
</html>
```

The HTML noscript Element

The <noscript> element is used to provide an alternate content for users that have either disabled JavaScript in their browser or have a browser that doesn't support client-side scripting.

This element can contain any HTML elements, aside from <script>, that can be included in the <body> element of a normal HTML page.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Noscript Demo</title>
</head>
<body>
  <div id="greet"></div>
  <script>
    document.getElementById("greet").innerHTML = "Hello World!";
  </script>
  <noscript>
    <p>Oops! This website requires a JavaScript-enabled browser.</p>
  </noscript>
</body>
```

</html>

Some characters are reserved in HTML, e.g. you cannot use the less than (<) or greater than (>) signs or angle brackets within your text, because the browser could mistake them for markup, while some characters are not present on the keyboard like copyright symbol ©.

Result	Description	Entity Name	Numerical reference
	non-breaking space		
<	less than	&lt;	&#60;
>	greater than	&gt;	&#62;
&	ampersand	&amp;	&#38;
"	quotation mark	&quot;	&#34;
'	apostrophe	&apos;	&#39;
¢	cent	&cent;	&#162;
£	pound	&pound;	&#163;
¥	yen	&yen;	&#165;
€	euro	&euro;	&#8364;
©	copyright	&copy;	&#169;
®	registered trademark	&reg;	&#174;
™	trademark	&trade;	&#8482;

Input Type Color

The color input type allows the user to select a color from a color picker and returns the color value in hexadecimal format (#rrggbb). If you don't specify a value, the default is #000000, which is black.

```
<form>
  <label for="mycolor">Select Color:</label>
  <input type="color" value="#00ff00" id="mycolor">
</form>
```

The date input type allows the user to select a date from a drop-down calendar.

The date value includes the year, month, and day, but not the time.

```
<form>
  <label for="mydate">Select Date:</label>
  <input type="date" value="2019-04-15" id="mydate">
</form>
```

The datetime-local input type allows the user to select both local date and time, including the year, month, and day as well as the time in hours and minutes.

```
<form>
  <label for="mydatetime">Choose Date and Time:</label>
  <input type="datetime-local" id="mydatetime">
</form>
```

The email input type allows the user to enter e-mail address. It is very similar to a standard text input type, but if it is used in combination with the required attribute, the browser may look for the patterns to ensure a properly-formatted e-mail address should be entered.

```
<form>
  <label for="myemail">Enter Email Address:</label>
  <input type="email" id="myemail" required>
</form>
```

The month input type allows the user to select a month and year from a drop-down calendar.

The value is a string in the format "YYYY-MM", where YYYY is the four-digit year and MM is the month number.

```
<form>
  <label for="mymonth">Select Month:</label>
  <input type="month" id="mymonth">
</form>
```

The number input type can be used for entering a numerical value. You can also restrict the user to enter only acceptable values using the additional attributes min, max, and step.

```
<form>
  <label for="mynumber">Enter a Number:</label>
  <input type="number" min="1" max="10" step="0.5" id="mynumber">
</form>
```

The range input type can be used for entering a numerical value within a specified range. It works very similar to number input, but it offers a simpler control for entering a number.

```
<form>
  <label for="mynumber">Select a Number:</label>
  <input type="range" min="1" max="10" step="0.5" id="mynumber">
</form>
```

The search input type can be used for creating search input fields.

A search field typically behaves like a regular text field, but in some browsers like Chrome and Safari as soon as you start typing in the search box a small cross appears on the right side of the field that lets you quickly clear the search field.

```
<form>
  <label for="mysearch">Search Website:</label>
  <input type="search" id="mysearch">
</form>
```

The tel input type can be used for entering a telephone number.

```
<form>
  <label for="myphone">Telephone Number:</label>
  <input type="tel" id="myphone" placeholder="xx-xxxx-xxxx" required>
</form>
```

The time input type can be used for entering a time (hours and minutes).

```
<form>
  <label for="mytime">Select Time:</label>
  <input type="time" id="mytime">
</form>
```

The url input type can be used for entering URL's or web addresses.

You can use the multiple attribute to enter more than one URL. Also, if required attribute is specified browser will automatically carry out validation to ensure that only text that matches the standard format for URLs is entered into the input box.

```
<form>
   <label for="myurl">Enter Website URL:</label>
   <input type="url" id="myurl" required>
</form>
```

The week input type allows the user to select a week and year from a drop-down calendar.

```
<form>
   <label for="myweek">Select Week:</label>
   <input type="week" id="myweek">
</form>
```

This section will explain you how to draw lines on canvas

Create a html page with following code

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Drawing on Canvas</title>
<script>
   window.onload = function() {
      var canvas = document.getElementById("myCanvas");
      var context = canvas.getContext("2d");
      // draw stuff here
   };
```

```
</script>
</head>
<body>
  <canvas id="myCanvas" width="300" height="200"></canvas>
</body>
</html>
```

Drawing a Line

The most basic path you can draw on canvas is a straight line. The most essential methods used for this purpose are moveTo(), lineTo() and the stroke().

The moveTo() method defines the position of drawing cursor onto the canvas, whereas the lineTo() method used to define the coordinates of the line's end point, and finally the stroke() method is used to make the line visible.

```
<script>
  window.onload = function() {
    var canvas = document.getElementById("myCanvas");
    var context = canvas.getContext("2d");
    context.moveTo(50, 150);
    context.lineTo(250, 50);
    context.stroke();
  };
</script>
```

Drawing a Arc

You can create arcs using the arc() method.

```
<script>
   window.onload = function() {
      var canvas = document.getElementById("myCanvas");
      var context = canvas.getContext("2d");
      context.arc(150, 150, 80, 1.2 * Math.PI, 1.8 * Math.PI, false);
      context.stroke();
   };
</script>
```

Drawing a Rectangle

You can create rectangle and square shapes using the rect() method. This method requires four parameters x, y position of the rectangle and its width and height.

```
<script>
        window.onload = function() {
      var canvas = document.getElementById("myCanvas");
      var context = canvas.getContext("2d");
      context.rect(50, 50, 200, 100);
      context.stroke();
   };
</script>
```

Drawing a Circle

There is no specific method for creating circle like rectangle's rect() method. However, you can create a fully enclosed arc such as circle using the arc() method.

```
context.arc(centerX, centerY, radius, 0, 2 * Math.PI, false);
```

```
<script>
        window.onload = function() {
    var canvas = document.getElementById("myCanvas");
    var context = canvas.getContext("2d");
    context.arc(150, 100, 70, 0, 2 * Math.PI, false);
    context.stroke();
  };
</script>
```

Applying Styles and Colors on Stroke

The default color of the stroke is black and its thickness is one pixel. But, you can set the color and width of the stoke using the strokeStyle and lineWidth property respectivley.

```
<script>
        window.onload = function() {
    var canvas = document.getElementById("myCanvas");
    var context = canvas.getContext("2d");
    context.lineWidth = 5;
    context.strokeStyle = "orange";
    context.moveTo(50, 150);
    context.lineTo(250, 50);
    context.stroke();
  };
</script>
```

Filling Colors inside Canvas Shapes

You can also fill color inside the canvas shapes using the fillStyle() method.

```
<script>
        window.onload = function() {
    var canvas = document.getElementById("myCanvas");
    var context = canvas.getContext("2d");
    context.rect(50, 50, 200, 100);
    context.fillStyle = "#FB8B89";
    context.fill();
    context.lineWidth = 5;
    context.strokeStyle = "black";
    context.stroke();
  };
</script>
```

Drawing Text on Canvas

You can also draw text onto canvas. These texts can contain any Unicode characters. The following example will draw a simple greeting message "Hello World!" onto a canvas.

```
<script>
        window.onload = function() {
    var canvas = document.getElementById("myCanvas");
    var context = canvas.getContext("2d");
    context.font = "bold 32px Arial";
    context.textAlign = "center";
    context.textBaseline = "middle";
    context.fillStyle = "orange";
    context.fillText("Hello World!", 150, 100);
```

```
};
</script>
```

Filling Gradient Colors inside Canvas Shapes

You can also fill gradient color inside the canvas shapes. A gradient is just a smooth visual transition from one color to another. There are two types of gradient available — linear and radial.

```
var grd = context.createLinearGradient(startX, startY, endX, endY);
```

The following example uses the createLinearGradient() method to fill a linear gradient color inside a rectangle

```
<script>
        window.onload = function() {
    var canvas = document.getElementById("myCanvas");
    var context = canvas.getContext("2d");
    context.rect(50, 50, 200, 100);
    var grd = context.createLinearGradient(0, 0, canvas.width, canvas.height);
    grd.addColorStop(0, '#8ED6FF');
    grd.addColorStop(1, '#004CB3');
    context.fillStyle = grd;
    context.fill();
    context.stroke();
};
</script>
```

What is SVG?

The Scalable Vector Graphics (SVG) is an XML-based image format that is used to define two-dimensional vector based graphics for the web. Unlike raster image (e.g. .jpg, .gif, .png, etc.), a vector image can be scaled up or down to any extent without losing the image quality.

An SVG image is drawn out using a series of statements that follow the XML schema — that means SVG images can be created and edited with any text editor, such as Notepad. There are several other advantages of using SVG over other image formats like JPEG, GIF, PNG, etc.

SVG images can be searched, indexed, scripted, and compressed.

SVG images can be created and modified using JavaScript in real time.

SVG images can be printed with high quality at any resolution.

SVG content can be animated using the built-in animation elements.

SVG images can contain hyperlinks to other documents.

Embedding SVG into HTML Pages

You can embed SVG graphics directly into your document using the HTML5 <svg> element.

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="utf-8">
   <title>Embedding SVG in HTML</title>
</head>
<body>
   <svg width="300" height="200">
      <text x="10" y="20" style="font-size:14px;">
         Your browser support SVG.
      </text>
      Sorry, your browser does not support SVG.
   </svg>
</body>
</html>
```

Drawing Path and Shapes with SVG

The following section will explain you how to draw basic vector-based paths and shapes on the web pages using the newly introduced HTML5 <svg> element.

Drawing a Line

The most basic path you can draw with SVG is a straight line. The following example will show you how to create a straight line using the SVG <line> element:

The attributes x1, x2, y1 and y2 of the <line> element draw a line from (x1,y1) to (x2,y2).

```
<svg width="300" height="200">
  <line x1="50" y1="50" x2="250" y2="150" style="stroke:red; stroke-width:3;" />
</svg>
```

Drawing a Rectangle

You can create simple rectangle and square shapes using the SVG <rect> element. The following example will show you how to create and style a rectangular shape with SVG:

The attributes x and y of <rect> element defines the co-ordinates of the top-left corner of the rectangle. The attributes width and height specifies the width and height of the shape.

```
<svg width="300" height="200">
  <rect x="50" y="50" width="200" height="100" style="fill:orange; stroke:black; stroke-width:3;" />
</svg>
```

Drawing a Circle

You can also create the circle shapes using the SVG <circle> element. The following example will show you how to create and style a circular shape with SVG:

The attributes cx and cy of the <circle> element defines the co-ordinates of the center of the circle and the attribute r specifies the radius of the circle. However, if the attributes cx and cy are omitted or not specified, the center of the circle is set to (0,0).

```
<svg width="300" height="200">
  <circle cx="150" cy="100" r="70" style="fill:lime; stroke:black; stroke-width:3;" />
</svg>
```

Drawing Text with SVG

You can also draw text on the web pages with SVG. The text in SVG is rendered as a graphic so you can apply all the graphic transformation to it but it is still acts like text — that means it can be selected and copied as text by the user.

The attributes x and y of the <text> element defines the location of the top-left corner in absolute terms whereas the attributes dx and dy specifies the relative location.

You can even use the <tspan> element to reformat or reposition the span of text contained within a <text> element. Text contained in separate tspans, but inside the same text element can all be selected at once — when you click and drag to select the text. However, the text in separate text elements cannot be selected at the same time.

```
<svg width="300" height="200">
  <text x="20" y="30" style="fill:purple; font-size:22px;">
    Welcome to Our Website!
  </text>
  <text x="20" y="30" dx="0" dy="20" style="fill:navy; font-size:14px;">
    Here you will find lots of useful information.
  </text>
</svg>
```

Embedding Audio in HTML Document

Inserting audio onto a web page was not easy before, because web browsers did not have a uniform standard for defining embedded media files like audio.

In this chapter we'll demonstrates some of the many ways to embed sound in your webpage, from the use of a simple link to the use of the latest HTML5 <audio> element.

Using the HTML5 audio Element

The newly introduced HTML5 <audio> element provides a standard way to embed audio in web pages. However, the audio element is relatively new but it works in most of the modern web browsers.

The following example simply inserts an audio into the HTML5 document, using the browser default set of controls, with one source defined by the src attribute.

```
<audio controls="controls" src="media/birds.mp3">
   Your browser does not support the HTML5 Audio element.
</audio>
```

An audio, using the browser default set of controls, with alternative sources.

```
<audio controls="controls">
   <source src="media/birds.mp3" type="audio/mpeg">
   <source src="media/birds.ogg" type="audio/ogg">
   Your browser does not support the HTML5 Audio element.
</audio>
```

Linking Audio Files

You can make links to your audio files and play it by ticking on them.

```
<a href="media/sea.mp3">Track 1</a>
<a href="media/wind.mp3">Track 2</a>
```

Using the object Element

The <object> element is used to embed different kinds of media files into an HTML document. Initially, this element was used to insert ActiveX controls, but according to the specification, an object can be any media object such as audio, video, PDF files, Flash animations or even images.

```
<object data="media/sea.mp3"></object>

<object data="media/sea.ogg"></object>
```

Using the embed Element

The <embed> element is used to embed multimedia content into an HTML document.

```
<embed src="media/wind.mp3">

<embed src="media/wind.ogg">
```

Embedding Video in HTML Document

Inserting video onto a web page was not relatively easy, because web browsers did not have a uniform standard for defining embedded media files like video.

In this chapter we'll demonstrates some of the many ways of adding videos on web pages, from the latest HTML5 <video> element to the popular YouTube videos.

Using the HTML5 video Element

The newly introduced HTML5 <video> element provides a standard way to embed video in web pages. However, the video element is relatively new, but it works in most of the modern web browsers.

The following example simply inserts a video into the HTML document, using the browser default set of controls, with one source defined by the src attribute.

```
<video controls="controls" src="media/shuttle.mp4">
    Your browser does not support the HTML5 Video element.
</video>

<video controls="controls">
```

```
    <source src="media/shuttle.mp4" type="video/mp4">
    <source src="media/shuttle.ogv" type="video/ogg">
    Your browser does not support the HTML5 Video element.
</video>
```

Using the object Element

The <object> element is used to embed different kinds of media files into an HTML document. Initially, this element was used to insert ActiveX controls, but according to the specification, an object can be any media object such as video, audio, PDF files, Flash animations or even images.

The following code fragment embeds a Flash video into a web page.

```
<object data="media/blur.swf" width="400px" height="200px"></object>
```

Using the embed Element

The <embed> element is used to embed multimedia content into an HTML document.

The following code fragment embeds a Flash video into a web page.

```
<embed src="media/blur.swf" width="400px" height="200px">
```

Embedding the YouTube Videos

This is the easiest and popular way to embed videos files in the web pages. Just upload the video on YouTube and insert HTML code to display that video in your web page.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>YouTube Video</title>
</head>
<body>
```

```
  <iframe width="560" height="315" src="//www.youtube.com/embed/YE7VzlLtp-4" frameborder="0" allowfullscreen></iframe>

</body>

</html>
```

What is Web Storage?

The HTML5's web storage feature lets you store some information locally on the user's computer, similar to cookies, but it is faster and much better than cookies. However, web storage is no more secure than cookies. Please check out the tutorial on PHP cookies to learn more about cookies.

The information stored in the web storage isn't sent to the web server as opposed to the cookies where data sent to the server with every request. Also, where cookies let you store a small amount of data (nearly 4KB), the web storage allows you to store up to 5MB of data.

There are two types of web storage, which differ in scope and lifetime:

Local storage — The local storage uses the localStorage object to store data for your entire website on a permanent basis. That means the stored local data will be available on the next day, the next week, or the next year unless you remove it.

Session storage — The session storage uses the sessionStorage object to store data on a temporary basis, for a single browser window or tab. The data disappears when session ends i.e. when the user closes that browser window or tab.

The localStorage Object

As stated earlier, the localStorage object stores the data with no expiration date. Each piece of data is stored in a key/value pair. The key identifies the name of the information (like 'first_name'), and the value is the value associated with that key (say 'Peter'). Here's an example:

```
<script>

// Check if the localStorage object exists

if(localStorage) {

  // Store data

  localStorage.setItem("first_name", "Peter");


  // Retrieve data

  alert("Hi, " + localStorage.getItem("first_name"));
```

```
} else {
   alert("Sorry, your browser do not support local storage.");
}
</script>
```

Example explained:

The above JavaScript code has the following meaning:

localStorage.setItem(key, value) stores the value associated with a key.

localStorage.getItem(key) retrieves the value associated with the key.

You can also remove a particular item from the storage if it exists, by passing the key name to the removeItem() method, like localStorage.removeItem("first_name").

However, if you want to remove the complete storage use the clear() method, like localStorage.clear(). The clear() method takes no arguments, and simply clears all key/value pairs from localStorage at once, so think carefully before you using it.

The sessionStorage Object

The sessionStorage object work in the same way as localStorage, except that it stores the data only for one session i.e. the data remains until the user closes that window or tab.

Let's try out the following example to understand how it basically works:

```
<script>
// Check if the sessionStorage object exists
if(sessionStorage) {
   // Store data
   sessionStorage.setItem("last_name", "Parker");

   // Retrieve data
   alert("Hi, " + localStorage.getItem("first_name") + " " + sessionStorage.getItem("last_name"));
} else {
```

```
    alert("Sorry, your browser do not support session storage.");
}
</script>
```

HTML5 Geolocation

The HTML5 geolocation feature lets you find out the geographic coordinates (latitude and longitude numbers) of the current location of your website's visitor.

This feature is helpful for providing better browsing experience to the site visitor. For example, you can return the search results that are physically close to the user's location.

Finding a Visitor's Coordinates

Getting the position information of the site visitor using the HTML5 geolocation API is fairly simple. It utilizes the three methods that are packed into the navigator.geolocation object — getCurrentPosition(), watchPosition() and clearWatch().

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Get Current Position</title>
<script>
    function showPosition() {
        if(navigator.geolocation) {
            navigator.geolocation.getCurrentPosition(function(position) {
                var positionInfo = "Your current position is (" + "Latitude: " + position.coords.latitude + ", " + "Longitude: " + position.coords.longitude + ")";
                document.getElementById("result").innerHTML = positionInfo;
            });
        } else {
            alert("Sorry, your browser does not support HTML5 geolocation.");
        }
    }
```

```
</script>
</head>
<body>
  <div id="result">
    <!--Position information will be inserted here-->
  </div>
  <button type="button" onclick="showPosition();">Show Position</button>
</body>
</html>
```

Dealing with Errors and Rejections

There may be a situation when a user does not want to share his location data with you. To deal with such situations, you can supply two functions when you call the getCurrentLocation() function.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Handling Geolocation Errors</title>
<script>
  // Set up global variable
  var result;

  function showPosition() {
    // Store the element where the page displays the result
    result = document.getElementById("result");

    // If geolocation is available, try to get the visitor's position
    if(navigator.geolocation) {
      navigator.geolocation.getCurrentPosition(successCallback, errorCallback);
      result.innerHTML = "Getting the position information...";
```

```
    } else {
        alert("Sorry, your browser does not support HTML5 geolocation.");
    }
};

// Define callback function for successful attempt
function successCallback(position) {
    result.innerHTML = "Your current position is (" + "Latitude: " + position.coords.latitude + ", " +
"Longitude: " + position.coords.longitude + ")";
}

// Define callback function for failed attempt
function errorCallback(error) {
    if(error.code == 1) {
        result.innerHTML = "You've decided not to share your position, but it's OK. We won't ask you
again.";
    } else if(error.code == 2) {
        result.innerHTML = "The network is down or the positioning service can't be reached.";
    } else if(error.code == 3) {
        result.innerHTML = "The attempt timed out before it could get the location data.";
    } else {
        result.innerHTML = "Geolocation failed due to unknown error.";
    }
}
</script>
</head>
<body>
<div id="result">
    <!--Position information will be inserted here-->
</div>
<button type="button" onclick="showPosition();">Show Position</button>
</body>
```

```
</html>
```

Showing Location on Google Map

You can do very interesting things with geolocation data, like showing the user location on Google map. The following example will show your current location on Google map based the latitude and longitude data retrieved through the HTML5 geolocation feature.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Using the Google Maps</title>
<script>
   function showPosition() {
      navigator.geolocation.getCurrentPosition(showMap);
   }

   function showMap(position) {
      // Get location data
      var latlong = position.coords.latitude + "," + position.coords.longitude;

      // Set Google map source url
      var mapLink =
"https://maps.googleapis.com/maps/api/staticmap?center="+latlong+"&zoom=16&size=400x300&o
utput=embed";

      // Create and insert Google map
      document.getElementById("embedMap").innerHTML = "<img alt='Map Holder' src='"+ mapLink
+"'>";
   }
</script>
</head>
<body>
   <button type="button" onclick="showPosition();">Show My Position on Google Map</button>
```

```
    <div id="embedMap">
        <!--Google map will be embedded here-->
    </div>
</body>
</html>
```