**HTML5**

* HTML5 is the latest and most enhanced version of HTML
* HTML5 introduces a number of new elements and attributes that helps in building a modern website

**New Features**

* **New Semantic Elements :** (clearly describes meaning to browser and to the developer)
  + <header>, <footer>, and <section> which clearly defines the content which the respective element holds, unlike non semantic elements like <div>, <span>
* **Web Forms 2.0 :** 
  + Improvements to HTML web forms where new attributes have been introduced for <input> tag
* **Storage in html5**
* **Canvas :** 
  + two-dimensional drawing surface that can be programed using JavaScript
* **Audio & Video :** 
  + Allows you to embed audio or video on your web pages
* **Drag and drop:** 
  + Drag and drop the items from one location to another location on a the same webpage

**Semantic Markup**

* Gives a meaning to tags
* For example, we use <div> element to create header, footer, place a particular content in page etc. but now with the semantic tags we have different tags to fit each purpose.
* Doctype greatly simplified
  + <!DOCTYPE html>
* Meta tags  
  <meta name="keywords" content="html5, css3, javascript, angular">  
  <meta name="description" content="Training on front end web technologies">

**Structural Elements**

* **<header> & <hgroup>**
  + If you want to group header and a sub header inside a <header> tag then it can be grouped using <hgroup>
  + **<hgroup>  
     <header>  
     <h1>Header 1</h1>  
     <h2>Sub header</h2>  
     </header>  
    </hgroup>**
* **Section:** This tag represents a generic document or application section. It can be used together with h1-h6 to indicate the document structure.  
  + <section>  
     <h1>Cricket</h1>  
     <p>Cricket is a bat-and-ball game played between two teams of 11 players each on a field at the centre of which is a rectangular 22-yard-long pitch</p>  
    </section>
* **article :** This tag represents an independent piece of content of a document, such as a blog entry or newspaper article  
    
  + <article>  
     <h1>Cricket</h1>  
     <p>Cricket is a bat-and-ball game played between two teams of 11 players each on a field at the centre of which is a rectangular 22-yard-long pitch</p>  
    </article>
* **header :** This tag represents the header of a section.
  + <article>  
     <header>  
     <h1>Cricket</h1>  
     <p>From Wikipedia, the free encyclopedia</p>  
     </header>  
     <p>Cricket is a bat-and-ball game played between two teams of 11 players each on a field at the centre of which is a rectangular 22-yard-long pitch. The game is played by 120 million players in many countries, making it the world's second most popular sport after association football.[1][2][3] Each team takes its turn to bat, attempting to score runs, while the other team fields. Each turn is known as an innings</p>  
    </article>
* **footer** : This tag represents a footer for a section and can contain information about the author, copyright information, etc.
  + <footer>  
     <p>Posted by: Arungopan</p>  
     <p>Contact: <a href="mailto:arun.gopan@marlabs.com">  
     arun.gopan@marlabs.com</a>.</p>  
    </footer>
* **Nav :** This tag represents a section of the document intended for navigation.
  + <nav>  
     <a href="/cricket/">Cricket</a> |  
     <a href="/football/">Football</a> |  
     <a href="/tennis/">Tennis</a> |  
     <a href="/baseball/">Baseball</a>  
    </nav>
* **aside :** This tag represents a piece of content that is only slightly related to the rest of the page.
  + <p>Me and my family visited Malaysia last year.</p>  
      
    <aside>  
     <h4>Twin tower, Malayasia</h4>  
     <p>The Petronas Towers, also known as the Petronas Twin Towers (Malay: Menara Petronas, or Menara Berkembar Petronas), are twin skyscrapers in Kuala Lumpur, Malaysia. According to the Council on Tall Buildings and Urban Habitat (CTBUH)'s official definition and ranking, they were the tallest buildings in the world from 1998 to 2004 and remain the tallest twin towers in the world. </p>  
    </aside>
* **figure :** This tag can be used to associate a caption together with some embedded content, such as a graphic or video.
  + <figure>  
     <img src="https://upload.wikimedia.org/wikipedia/commons/thumb/6/6d/Petronas\_Twin\_Towers\_2010\_April.jpg/250px-Petronas\_Twin\_Towers\_2010\_April.jpg" alt="The Pulpit Rock" width="304" height="228">  
     <figcaption>Petronas Towers, Malaysia.</figcaption>  
    </figure>

**Attributes**

* As in HTML, elements may contain attributes that are used to set various properties of an element.
* Some of the standard attributes
  + Align
  + Bgcolor
  + Background
  + Class
  + Data-attrname
  + Height
  + Width
* **Custom Attributes:**
  + A custom attribute starts with data-
  + <div class="example" data-subject="physics" data-level="complex">  
     ...  
    </div>

**Web Form 2.0**

* By introducing new form elements and attributes HTML5 removes the need for tedious scripting and styling that was required in previous version of HTML
* New HTML5 input elements includes the following
  + Date
    - <input type=”date” name=”date\_input” />
  + Month
    - <input type=”month” name=”month\_input” />
  + Week
    - <input type=”week” name=”week\_input” />
  + Time
    - <input type=”time” name=”time\_input” />
  + Number
    - <input type=”number” min=”10” max=”100” step=”5" name=”number\_input” />
  + Range
    - <input type=”range” min=”10” max=”100” />
  + Email
    - <input type=”email” name=”email\_input” />
  + Url
    - <input type=”url” name=”url\_input” />
  + Search
    - <input type=”search” />
* **Placeholder attribute:**
  + <input type="text" name="search" placeholder="Marlabs"/>
* **Autofocus attribute :**
  + <input type="text" name="username" autofocus />
* **Required attribute :**
  + <input type="text" name="username" required />
* **Multiple attribute**
  + <input type=”email” name=”email\_input” required multiple />
* **Datalist**
  + <input type="text" list="myDataList" />  
     <datalist id="myDataList">  
     <option value="Mr">Mr</option>  
     <option value="Mrs">Mrs</option>  
     <option value="Ms">Ms</option>  
     </datalist>
* **Form property**
  + Used to relate an input element to a form if placed outside the form
  + Use id of the form as the value for form attribute
  + <input type=”text” form=”myForm” />

**HTML GRAPHICS:** This portion talks about CANVAS / SVG elements

**CANVAS ELEMENT :**

* Canvas creates a drawing surface, a blank area within the browser.
* Uses javascript to draw graphics
  + <canvas id="canvas\_elem" width="200" height="200"></canvas>
* To find the canvas element from your page you could use
  + var canvas = document.getElementById("canvas\_elem");
* Example for simple canvas element:
  + <!DOCTYPE HTML>  
    <html>  
     <head>  
       
     <style>  
     #canvas\_elem{border:2px solid #FF0000;}  
     </style>  
       
     </head>  
     <body>  
      
     <canvas id="canvas\_elem" width="100" height="100"></canvas>  
       
     </body>  
    </html>
* **Browser support :**
  + We could use DOM method **getContext** of canvas element to check which rendering context and to browser compatability
    - var canvas = document.getElementById("canvas\_elem");
    - if (canvas.getContext){   
       var canvas\_context = canvas.getContext('2d');   
      }   
        
      else {   
       // Sorry your browser does not support canvas  
      }
* **Rectangles:**
  + Three methods for drawing rectangles  
    - fillRect(x,y,width,height)
      * Draws a filled rectangle
    - strokeRect(x,y,width,height)
      * Draws a rectangular outline
    - clearRect(x,y,width,height)
      * Make a fully transparent rectangle
    - fillStyle : specify color code to fill the rectangle
  + Example :
    - <html>  
      <body>  
      <canvas id="canvas\_elem" style="border:1px solid #ff0000" width="200" height="200"></canvas>  
        
      <input type="button" id="button1" onclick="draw\_rect()" value="Draw rectangle" />  
      </body>  
      <script>  
      function draw\_rect() {  
       var canvas\_elem = document.getElementById("canvas\_elem");  
       if(canvas\_elem.getContext) {  
       var canvas\_ctx = canvas\_elem.getContext("2d");  
       canvas\_ctx.fillStyle = "#ff0000";  
       canvas\_ctx.fillRect(10,10, 150, 150);  
       }  
        
      }  
      </script>  
      </html>
* **Line:**
  + Methods available
    - moveTo(x,y)
    - lineTo(x,y)
    - stroke() : draws the path that we define using moveTo, lineTo etc functions
  + Examples
    - Creates alphabet “W”  
      <canvas id="canvas\_elem" width="300" height="150" style="border:1px solid #d3d3d3;"></canvas>  
        
      var c = document.getElementById("canvas\_elem");  
      var canvas\_ctx = c.getContext("2d");  
      canvas\_ctx.beginPath();  
      canvas\_ctx.moveTo(40, 40);  
      canvas\_ctx.lineTo(60, 100);  
      canvas\_ctx.lineTo(80, 60);  
      canvas\_ctx.lineTo(100, 100);  
      canvas\_ctx.lineTo(120, 40);  
      canvas\_ctx.strokeStyle = "red";  
      canvas\_ctx.stroke();
* **Circle:**
  + Methods available
    - arc( x-coordinate of the center of the circle, y-coordinate of the center of the circle, radius, starting angle, in radians, ending angle)
  + Example
    - var c = document.getElementById("canvas\_elem");  
      var canvas\_ctx= c.getContext("2d");  
      canvas\_ctx.arc(200,150,100,0,2);  
      canvas\_ctx.stroke();

**SVG ELEMENT**

* The SVG Element is used for graphical representation with HTML5
* It defines the Vector based graphics
* vector graphics are comprised of paths, which are defined by a start and end point, along with other points, curves, and angles
* Since vector graphics are not made up of a grid of pixels, they can be scaled to a larger size and not lose any image quality
* **<svg>** tag is the container for SVG graphics
* Examples
  + Creating a rectangle  
      
      
    <svg width="400" height="100">  
     <rect x="50" y="20" rx="20" ry="20" width="150" height="150" fill="red" stroke-width="3" stroke="#000" /></svg>
* Circles
  + <circle cx="50" cy="50" r="40" stroke="#FF0000" stroke-width="3" fill="#EEE" />
* Ellipse :
  + <ellipse cx="200" cy="80" rx="100" ry="50" fill="red" stroke-width="3" stroke="#000" />
  + Rx : horizontal radius
  + Ry : vertical radius
* Line :
  + <line x1="50" y1="50" x2="200" y2="200" fill="red" stroke-width="3" stroke="#000" />
* Polygon :
  + <polygon points="200,10 400,10 400,400, 200,400" fill="red" stroke-width="3" stroke="#000" />
  + <polygon points="200,10 250,200 150,200" fill="red" stroke-width="3" stroke="#000" />

**Storage in HTML5**

* HTML5 offers two ways by which applications can store data in client browsers
  + Session Storage
  + Local Storage
* Before using storage in HTML5, we need to confirm whether storage is available
  + if (typeof(Storage) !== "undefined") {  
     //write you code here  
    } else {  
     // error  
    }
* **Session Storage:**
  + Stores data for a particular session and the data will be lost once the user the specific browser tab / window
  + Examples
    - sessionStorage.name = “Arungopan”; // sets the value “Arungopann” to index name of sessionStorage object
    - Var name = sessionStorage.name; // will fetch the value of name from sessionStorage object
* Local storage
  + Stores data without any expiry.
  + Data will available across multiple sessions.
  + Examples
    - localStorage.name = “Arun”;
    - Var name = localStorage.name;
* One could also use methods “**setItem(key, value)**, **getItem(key)**, **removeItem(key)** ” to store / fetch / remove informations stored in storage

**Embedding Audio / Video content**

* HTML5 allows you to embed audio / video content in your web page with the help of **<video>, <audio>** tags
* <video src="marlabs.mp4" width="500" height="300"></video>
* With **<source>** tag you could specify multiple source elements from which the browser will select the first recognized format
  + <video width="500" height="300" controls autoplay>  
     <source src="marlabs.mp4" type="video/ogg" />  
     <source src="marlabs.mp4" type="video/mp4" />  
     The browser which you currently using does not support video playback  
     </video>
  + Formats supported are mpeg4, Ogg
* <audio src="marlabs.mp3" controls autoplay></audio>
* <audio controls autoplay>  
   <source src="marlabs.wav" type="audio/wav" />  
   <source src="marlabs.mp3" type="audio/mp3" />  
   Your browser does not support the audio element.  
   </audio>

**Drag and drop**

* draggable attribute
  + <img src=”image.jpg” draggable=”true” />
* Events
  + dragstart
  + dragover
  + Drop
* dataTransfer Object
  + Contains the data being sent during the drag and drop operations
  + Sets data in the dragstart event and read in the drop event

<div id="img1" class="drag\_img">

<img id="img1\_img" class="img1" src="http://www.topgear.com/india/images/stories/Car-Bike\_topImages/bike.jpg" /><br />

<span id="img1\_txt">Caption for img 1</span>

</div>

<div id="img2" class="drag\_img">

<img id="img2\_img" class="img1" src="http://www.topgear.com/india/images/stories/Car-Bike\_topImages/bike.jpg" /><br />

<span id="img2\_txt">Caption for img 2</span>

</div>

<div id="img3" class="drag\_img">

<img id="img3\_img" class="img1" src="http://www.topgear.com/india/images/stories/Car-Bike\_topImages/bike.jpg" /><br />

<span id="img3\_txt">Caption for img 3</span>

</div>

<br />

<div id="drop\_here" style="border:1px solid #000;padding:10px;min-height:100px;">

</div>

window.onload = function() {

var drag\_images = document.getElementsByClassName('drag\_img');

var drop\_cntr = document.getElementById('drop\_here');

var img\_len = drag\_images.length;

for(var i=0; i<img\_len; i++) {

drag\_images[i].addEventListener('dragstart', function(event) {

event.dataTransfer.setData('imgId', this.getAttribute('id'));

});

}

drop\_cntr.addEventListener('dragover', function(event) {

event.preventDefault();

return false;

});

drop\_cntr.addEventListener('drop', function(event) {

alert(event.dataTransfer.getData('imgId'));

});

};