HTML5

# Getting Started

* Need Notepad or webmatrix to write HTML5 pages. Any editor would suffice
* **<!DOCTYPE html>** //doctype element simplified for all revisions. 1st statement in html page
  + Ensures best a best –effort attempt to render. For legacy systems
* Other shorthand tags:
  + <html lang=”en”> //easy to specify language is english
  + <meta charset=”utf-8” /> //omit content type is text/html & http-equiv
  + <link rel=”stylesheet” href=”style.css”> //u can omit the type=”text/javascript”>
* Get the outline for page from <http://gsnedders.html5.org/outliner/>

# Structural Elements

* Semantic tags parse better and mirror reality
* **<hgroup> <header> <h1> title1</h1><h2>section title</h2></header></hgroup>**
  + header can be in section
* **<footer>** - more than 1 footer per page as each sectioning content can have a footer
  + Contains section info, author, copyright info
* **<article>** - self-contained documents like forum post, blog, article, email, comment
* **<section>** - not a stand alone document. Sections an article or page into subject area
* **<aside>** - refers to side bars, Nav sections ,pull quotes
* **<nav>** - used to mark major navigation sections not link, search result or sponsored links
* **<mark>** - highlight text relevant to user current activity
* **<time datetime= yyyy-mm-ddThh:mm-tz>** - date/time in universal format

# Changed Elements

* **<address>** - contact details (postal, email or others) of author & not postal address
  + Multiple in one document
* **<dl> <dt> term </dt> <dd> description </dd> </dl>** //name value groups
* **<em>** - change meaning of statement whereas **<i>** technical term or prose
* **<hr>** - horizontal rule
* **<ol start=n Reversed>** // starts order list @ n. Reversed not yet implemented.
* **<s>** - indicate strikethrough for no longer relevant / accurate content.
* **<small>** - small print as in legal documents
* **<strong>** - important whereas **<b>** - just bold

# Form Elements

* **<label for=”inputElementID”> Text </label>** - label associated for given input element
* **<input type=”text” autofocus>** //sets the focus on this control when page loads
  + For a telephone field we can include **pattern =”\(\d\d\d\) \d\d\d-\d\d\d\d”**
  + **tabindex=n** // allows to define the tab order in form
* **<input type=”date”/>** //displays a date picker. Implementation as per browser
* **<input type=”time”/>** //displays a time picker. Implementation as per browser
* **<input type=”month”/>** //displays month year. Implementation as per browser
* **<input type=”week”/>** //displays week (1-52) year. Implementation as per browser
* **<input type=”range” min=”x” max =”y”/>** // displays a plain bar
  + Use css3 to enhance

input[type=range]::after {content: attr(max);}

input[type=range]::before {content: attr(min);}

input[type=range] {width:250px; color:blue; font-family:’Arial’; font-size:small;}

* **<input type=”search” placeholder=”Enter search term”/>** // search field with placeholder property
* **<input type=”tel” required/>** //displays telephone number field which is required. No validation
* **<input type=”email” required multiple/>** //displays email field which is required.
  + Accept comma separated multiple values and basic email validation ex foo@bar.com
* **<input type=”url” />** //asks for a url that begins with http://
* **<input type=”color” />** //displays color picker.
* To display datalist we need a text box followed by datalist ex:
  + <input type=”text” list=”myDataList”>

<datalist id=”myDataList”>

<option label=”Mr” value=”Mr”></option>

<option label=”Mrs” value=”Mrs”></option>

<option label=”Miss” value=”Miss”></option>

</datalist> // allows to select & type so acts like combo box

* **<progress value=x max=y />** //displays progress bar showing x unit is completed out of y work units
* **<meter min=x max=y low=x1 high=y1 optimum=z value=z1>** // shows the meter bar alerting user when value is between (min & low) or (high & max).

# Media & Graphics Elements

Video & Audio

* IE & Safari supports only MP4 while firefox, chrome & opera supports WEBM.
* **<video src=”url” controls autoplay loop poster=”url” preload=”none| metadata | auto></video>**
  + **src** refers to the video url. Place **<source src”url”/>** multiple tags for ie & chrome.
  + **controls** display the video whereas **autoplay** display the video & fills the entire screen
  + **poster** tells the image url that is displayed initially.
  + **preload** specifies how to preload. Auto is default
* Set the web config for the video files
  + **<system.webServer><staticContent>**

**<mimeMap fileExtension=”.mp4” mimeType=”video/mp4” />**

**</staticContent></system.webServer>**

* IE & Safari supports only MP3
* **<audio src=”url” controls autoplay loop poster=”url” preload=”none|></audio>**

Canvas

* **<canvas id=”myCanvas” width=”500” height=”300”> </canvas>** // defines a 500 \* 300 drawing area
* **var elem=document.getElementByID(‘myCanvas’); //**get the canvas element
* **var canvasObject =elem.getContext(‘2d’) //** it can have either 2d or webgl
* **canvas.fillStyle=”hexcolor” or gradient object; //**sets the fill color. Default is black
* **canvas.StrokeStyle=”hexcolor”; //** sets the line color.Default is black
* **canvasObject.strokeRect(x,y,width,height) //**draws a rectangle outline
* **canvasObject.fillRect(x,y,width,height) //** draws a solid rectangle
* **canvasObject.clearRect(x,y,width,height) //** erases given rectangle area.
* **var gradient= canvasObject.createLinearGradient(x1,y1,x2,y2**) //creates a linear gradient for the canvas and not the shape
* **var gradient= canvasObject.createRadialGradient(x1,y1,r1,x2,y2,r2**) // creates radial gradient
* **gradient.addColorStop(x,color)** // x can be between 0-1
* Path is the instruction for pen to follow. They are send to canvas to make a drawing.
* **canvasObject.beginPath()** // begins the drawing of path on canvas
* **canvasObject.stroke()** // draws the actual path on canvas
* **canvasObject.fill()** // fill the path between the coordinates.
* **canvasObject.lineWidth=n;** //sets the width of the line
* **canvasObject.lineCap=”square|round”** // sets the cap of line
* **canvasObject .moveTo(x,y)** // moves the pen to given coordinates
* **canvasObject .lineTo(x,y)** // draws a line from current pen location to given coordinate
* **canvasObject closePath()** // draws a line from cuuent pen location to initial pen coordinates.
* **canvasObject .clip()** // clips the area that is specified.
* **canvasObject .rect(x,y,width,height)** // draws the rectangle of given width & height
* **canvasObject .arc(x,y,r,startAngle,endAngle,direction)** // draws an arc.
  + Angles are given in radian (Math.PI / 180 \*angle)
  + Direction can be true (anti-clock) and false (clock)
  + To make a circle set startAngle=0 & endAngle=Math.PI \*2
* **canvasObject .quadraticCurveTo(cpx,cpy,x,y)** // controls curve with 1 set of control points.
* **canvasObject .bezierCurveTo(cp1x,cp1y, cp2x,cp2y ,x,y)** // controls curve with 2 control points.
* **canvasObject.font=”bold 20px font1,font2”;** // sets the font values to be used to draw text canvas
  + Syntax is same as css
* **canvasObject.fillText(“string”,x,y);** // tells what is text to be put on canvas
* **canvasObject.textAlign=”start|end|left|right|center”** // horizontal alignment
* **.textBaseLine=”bottom|top|hanging|middle|alphabetic|ideographic”** //vertical alignment
* **canvasObject.measureText(string)** // takes the string and gives the height in pixels

# Storage in HTML5

Session Storage

* **sessionStorage[key]=value** // stores the key value pair in session storage like cookie
* **var value= sessionStorage[key]** // retrieves value based on the key.
* **sessionStorage .length** // gives number of items in the sessionStorage
* **sessionStorage .key(index)** // returns the key for the given index from session storage
* **sessionStorage.removeItem(key)** //deletes the given item from session storage
* **sessionStorage .clear()** // deletes all items from sessionStorage
* data is lost as soon as the session expires or window is closed

Local Storage

* **localStorage[key]=value** // stores the key value pair in session storage like cookie
* **var value= localStorage[key]** // retrieves value based on the key.
* **localStorage .length** // gives number of items in the sessionStorage
* **localStorage .key(index)** // returns the key for the given index from session storage
* **localStorage.removeItem(key)** //deletes the given item from session storage
* **localStorage .clear()** // deletes all items from sessionStorage
* **storage(e)** event // event is called when the changes are made to local storage
  + used to communicate between 2 instances of web application
  + the parameter e can be used to access **e.key,e.oldValue,e.newValue,e.url,e.storageArea**

# Cross Browser Handling

* Fallback Mechanism - Let user know if the browser supports the technology or not
  + **<audio> This browser does not support HTML5 </audio>**
* Feature Detection – checks for the elements method
  + **function supports\_canvas () {**

**return !!document.createElement(‘canvas’).getContext;**

**}** // returns true if canvas is created else returns false

* + Use **modernizer.custom.js** to check whether feature is supported or not. Provide the implementation in case of both. (**Modernizer.element** returns true if element is supported)