Introduction to HTML5

*Web Storage

*Web Storage

- HTML5 introduces two mechanisms, similar to HTTP session cookies, for storing structured data on the client side and to overcome following drawbacks.
  - Cookies are included with every HTTP request, thereby slowing down your web application by transmitting the same data.
  - Cookies are included with every HTTP request, thereby sending data unencrypted over the internet.
  - Cookies are limited to about 4 KB of data. Not enough to store required data.
- The two storages are session storage and local storage and they would be used to handle different situations.
- The latest versions of pretty much every browser supports HTML5 Storage including Internet Explorer.

*Web Storage

- localStorage and sessionStorage both extend Storage. There is no difference between them except for the intended "non-persistence" of sessionStorage.
- That is, the data stored in localStorage persists until explicitly deleted. Changes made are saved and available for all current and future visits to the site.
- For sessionStorage, changes are only available per tab. Changes made are saved and available for the current page in that tab until it is closed. Once it is closed, the stored data is deleted.

*Delete Web Storage

- Storing sensitive data on local machine could be dangerous and could leave a security hole.
- The Session Storage Data would be deleted by the browsers immediately after the session gets terminated.
- To clear a local storage setting you would need to call localStorage.remove('key'); where 'key' is the key of the value you want to remove. If you want to clear all settings, you need to call localStorage.clear() method.

Audio & Video

*Audio & Video

- HTML5 features include native audio and video support without the need for Flash.

- The HTML5 <audio> and <video> tags make it simple to add media to a website. You need to set src attribute to identify the media source and include a controls attribute so the user can play and pause the media.

Attribute	Description
src	Specifies the URL of the media source file
controls	Specifies whether or not to display media controls (such as a play/pause button etc)
autoplay	Specifies whether or not to start playing the media as soon as it has been loaded.
loop	Specifies whether to keep re-playing the media once it has finished.
poster=""	display a frame of the video (as a .jpg, .png..)
width=""	Specifies the width, in pixels, to display the video.
height=""	Specifies the height, in pixels, to display the video.

*Media Attributes

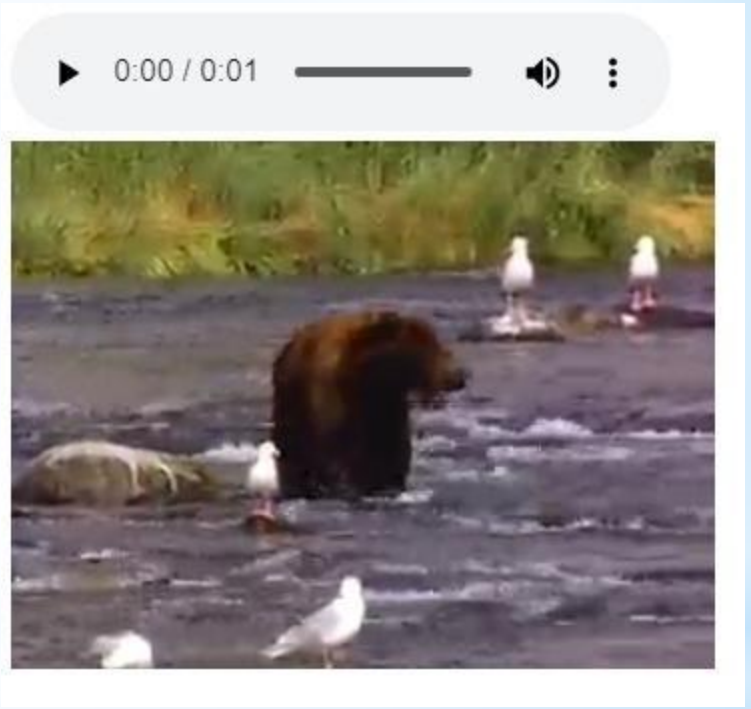
*Audio & Video

Audio Example

```
<audio controls autoplay autobuffer loop >
    <source src =
    "https://www.w3schools.com/html/horse.ogg" type
        = "audio/ogg" />
    <source src =
    "https://www.w3schools.com/html/horse.mp3" type
        = "audio/mpeg" />
      Your browser does not support the audio
    element.
</audio>
```

Video Example

```
<audio controls autoplay autobuffer loop >
    <source src =
    "https://www.w3schools.com/html/horse.ogg" type
        = "audio/ogg" />
    <source src =
    "https://www.w3schools.com/html/horse.mp3" type
        = "audio/mpeg" />
      Your browser does not support the audio
    element.
</audio>
```



controls : attribute will allow the user to control video playback.

autoplay : the media will automatically begin to play back as soon as it can do.

loop : the media will repeate automatically.

*Geolocation

*Geolocation

- HTML5 Geolocation API lets you share your location with your favorite web sites. A JavaScript can capture your latitude and longitude and can be sent to backend web server and do fancy location-aware things like finding local businesses or showing your location on a map.

- Today most of the browsers and mobile devices support Geolocation API. The geolocation APIs work with a new property of the global navigator object ie. Geolocation object

*Get Current Location

Get Current Location

```
navigator.geolocation.getCurrentPosition(showLocation, showError);
```

Show Location Date

```
function showLocation(position) {
    var latitude = position.coords.latitude;
    var longitude = position.coords.longitude;
    alert("Latitude : " + latitude + " Longitude: " + longitude);
}
```

Error Codes

```
function showError(error) {
  switch(error.code) {
    case error.PERMISSION_DENIED:
      alert("User denied the request for Geolocation.");
      break;
    case error.POSITION_UNAVAILABLE:
      alert("Location information is unavailable.");
      break;
    case error.TIMEOUT:
      alert("The request to get user location timed out.");
      break;
    case error.UNKNOWN_ERROR:
      alert("An unknown error occurred.");
      break;
  }
}
```

*Get Location Updates

Show Location Updates

```
var options = {timeout:60000};
watchID = navigator.geolocation.watchPosition(showLocation, errorHandler,
    options);
```

*Web RTC

*Web RTC

- Web RTC introduced by World Wide Web Consortium (W3C). That supports browser-tobrowser applications for voice calling, video chat, and P2P file sharing.
- If you want to try out? web RTC available for Chrome, Opera, and Firefox. Web RTC implements three API's as shown below −
- MediaStream − get access to the user's camera and microphone.
- RTCPeerConnection − get access to audio or video calling facility.
- RTCDataChannel − get access to peer-to-peer communication.

*Camera

Show camera Updates

```
//Video tag to display video
var video = document.querySelector("#videoElement");

    if (navigator.mediaDevices.getUserMedia) {
      navigator.mediaDevices.getUserMedia({ video: true })
        .then(function (stream) {
          video.srcObject = stream;
        })
        .catch(function (err0r) {
          console.log("Something went wrong!");
        });
    }
```

*SVG

* Raster / Pixel
* Vector

```
<SVG width="200" height="200"> </SVG>
```

* SVG

* Draw Rectangle using

```
<svg>
  <rect x="" y="" width=“" height=“"   style=“”>
</svg>
```

* Draw Line using

```
<svg>
  <line x1=“" y1=“” x2=“" y2=“‘ style=“”>
</svg>
```

*SVG

Canvas

Graphics “Canvas”

- Canvas is a new HTML element which can be used to draw graphics on a web page using Javascript.
- A canvas is a rectangular area, that you control every pixel of it.
- The canvas element has several methods for drawing paths, boxes, circles, characters, and adding images.

Graphics “Canvas”

* <canvas> element is an HTML tag, with the exception that its contents are rendered with JavaScript.
* It creates a fixed size drawing surface that exposes one or more rendering contexts using canvas context object.
* Each canvas element can only have one context that can be “2d”.

To draw a line using Canvas

* First, use the beginPath() method to declare that we are about to draw a new path.
* 2) Next, use the moveTo() method to position the context point (i.e. drawing cursor
* 3) Then, use the lineTo() method to draw a straight line from the starting position to a new position.
* 4) Finally, to make the line visible, we can apply a stroke to the line using stroke().

Drag and Drop

*Drag and Drop

*Drag and drop is a very common feature. It is when you "grab" an object and drag it to a different location.

```
<div id="mydiv" style="border:solid 1px red;
width:500px;height: 300px;"
ondragover="event.preventDefault();"
        ondrop="drop()">
        <p>Drag the image into the rectangle:</p>
    </div>
    <img src="img/lizard.png" draggable="true" alt=""
srcset="" ondragstart="drage(this)">
    <script>
        function drop() {
            console.log(item)
            document.getElementById('mydiv').append(it
em)
        }
        function drage(element) {
            item = element;
        }
        let item;
*
    </script>
```