HTML5 DRAG AND DROP

Drag and Drop is a very interactive and user-friendly concept that makes it easier to move an object to a different location by grabbing it. This allows the user to click and hold the mouse button over an element, drag it to another location, and release the mouse button to drop the element there. In HTML5 Drag and Drop are much easier to code and any element in it is draggable.

Drag and Drop Events:

Events	Description
ondrag	It is used to use when the element or text selection is being dragged in HTML.
ondragstart	It is used to call a function, drag(event), that specifies what data to be dragged
ondragenter	It is used to determine whether or not the drop target is to accept the drop. If the drop is to be accepted, then this event has to be canceled.
ondragleave	It occurs when the mouse leaves an element before a valid drop target while the drag is occurring.
ondragover	It specifies where the dragged data can be dropped.
ondrop	It specifies where the drop has occurred at the end of the drag operation.

Events	Description
ondragend	It occurs when the user has finished dragging an element.

Approach for Drag and Drop

- Set an element as draggable with the draggable attribute: <img draggable="true">.
- Specify the drag behavior using the ondragstart attribute, calling a function (drag(event)) to define the data to be dragged with event.dataTransfer.setData().
- Utilize the ondragover event to determine where the data can be dropped. Prevent default behavior with event.preventDefault() to enable the drop.
- Implement the ondrop event to perform the actual drop action

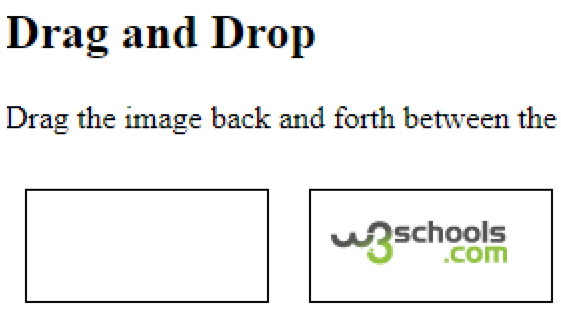
HTML Drag and Drop Example

The example below is a simple drag and drop example:

```
<!DOCTYPE HTML>

<html>

<head>

<style>

#div1, #div2 {

  float: left;

  width: 100px;

  height: 35px;

  margin: 10px;

  padding: 10px;

  border: 1px solid black;
}
</style>
<script>
function allowDrop(ev) {
  ev.preventDefault();
}
function drag(ev) {
  ev.dataTransfer.setData("text", ev.target.id);
}
function drop(ev) {
  ev.preventDefault();
  var data = ev.dataTransfer.getData("text");
  ev.target.appendChild(document.getElementById(data));
}
</script>
</head>
<body>
<h2>Drag and Drop</h2>
<p>Drag the image back and forth between the two div elements.</p>
<div id="div1" ondrop="drop(event)" ondragover="allowDrop(event)">
  <img src="img_w3slogo.gif" draggable="true" ondragstart="drag(event)" id="drag1"
width="88" height="31">
</div>
```

```
<div id="div2" ondrop="drop(event)" ondragover="allowDrop(event)"></div>

</body>

</html>
```

Output:



Drag and Drop

Drag the image back and forth between the two div elements.

HTML 5 AUDIO VIDEO CONTROLS

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.

Embedding Video

Here is the simplest form of embedding a video file in your webpage −

```
<video src = "foo.mp4"  width = "300" height = "200" controls>
            Your browser does not support the <video> element.
</video>
```

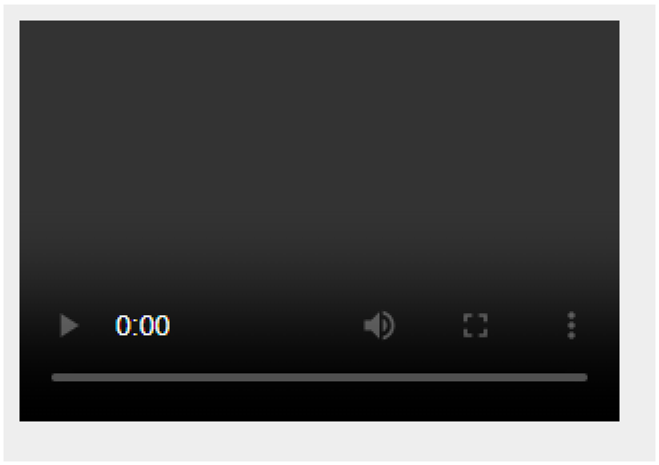
The current HTML5 draft specification does not specify which video formats browsers should support in the video tag. But most commonly used video formats are −

- **Ogg** − Ogg files with Thedora video codec and Vorbis audio codec.
- **mpeg4** − MPEG4 files with H.264 video codec and AAC audio codec.

You can use <source> tag to specify media along with media type and many other attributes. A video element allows multiple source elements and browser will use the first recognized format −

```
<!DOCTYPE HTML>
<html>
  <body>
    <video  width = "300" height = "200" controls autoplay>
      <source src = "/html5/foo.ogg" type ="video/ogg" />
      <source src = "/html5/foo.mp4" type = "video/mp4" />
      Your browser does not support the <video> element.
    </video>

  </body>
</html>
```

This will produce the following result –



Video Attribute Specification

The HTML5 video tag can have a number of attributes to control the look and feel and various functionalities of the control −

Sr.No.	Attribute & Description
1	**autoplay** This Boolean attribute if specified, the video will automatically begin to play back as soon as it can do so without stopping to finish loading the data.
2	**autobuffer** This Boolean attribute if specified, the video will automatically begin buffering even if it's not set to automatically play.
3	**Controls** If this attribute is present, it will allow the user to control video playback, including volume, seeking, and pause/resume playback.
4	**height** This attribute specifies the height of the video's display area, in CSS pixels.
5	**Loop** This Boolean attribute if specified, will allow video automatically seek back to the start after reaching at the end.
6	**preload** This attribute specifies that the video will be loaded at page load, and ready to run. Ignored if autoplay is present.
7	**Poster** This is a URL of an image to show until the user plays or seeks.
8	**src** The URL of the video to embed. This is optional; you may instead use the <source> element within the video block to specify the video to embed.

9	**Width** This attribute specifies the width of the video's display area, in CSS pixels.

Embedding Audio

HTML5 supports <audio> tag which is used to embed sound content in an HTML or XHTML document as follows.

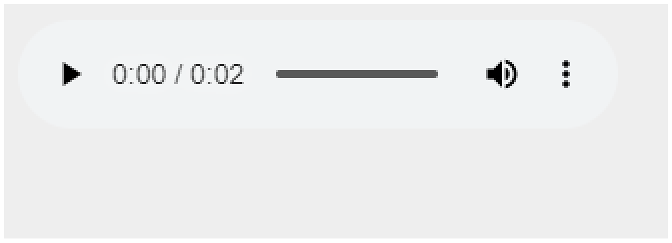
```
<audio src = "foo.wav" controls autoplay>
          Your browser does not support the <audio> element.
</audio>
```

The current HTML5 draft specification does not specify which audio formats browsers should support in the audio tag. But most commonly used audio formats are **ogg, mp3** and **wav**.

You can use <source&ggt; tag to specify media along with media type and many other attributes. An audio element allows multiple source elements and browser will use the first recognized format –

```
<!DOCTYPE HTML>

<html>
  <body>

    <audio controls autoplay>
      <source src = "/html5/audio.ogg" type = "audio/ogg" />
      <source src = "/html5/audio.wav" type = "audio/wav" />
      Your browser does not support the <audio> element.
    </audio>

  </body>
</html>
```

This will produce the following result −



Audio Attribute Specification

The HTML5 audio tag can have a number of attributes to control the look and feel and various functionalities of the control −

Sr.No.	Attribute & Description
1	**autoplay** This Boolean attribute if specified, the audio will automatically begin to play back as soon as it can do so without stopping to finish loading the data.
2	**autobuffer** This Boolean attribute if specified, the audio will automatically begin buffering even if it's not set to automatically play.
3	**controls** If this attribute is present, it will allow the user to control audio playback, including volume, seeking, and pause/resume playback.
4	**loop** This Boolean attribute if specified, will allow audio automatically seek back to the start after reaching at the end.
5	**preload** This attribute specifies that the audio will be loaded at page load, and ready to run. Ignored if autoplay is present.

6	**Src** The URL of the audio to embed. This is optional; you may instead use the <source> element within the video block to specify the video to embed.

Handling Media Events

The HTML5 audio and video tag can have a number of attributes to control various functionalities of the control using JavaScript −

S.No.	**Event & Description**
1	**abort** This event is generated when playback is aborted.
2	**canplay** This event is generated when enough data is available that the media can be played.
3	**ended** This event is generated when playback completes.
4	**Error** This event is generated when an error occurs.
5	**loadeddata** This event is generated when the first frame of the media has finished loading.
6	**Loadstart** This event is generated when loading of the media begins.
7	**pause** This event is generated when playback is paused.
8	**play**

	This event is generated when playback starts or resumes.
9	**progress** This event is generated periodically to inform the progress of the downloading the media.
10	**ratechange** This event is generated when the playback speed changes.
11	**seeked** This event is generated when a seek operation completes.
12	**seeking** This event is generated when a seek operation begins.
13	**suspend** This event is generated when loading of the media is suspended.
14	**volumechange** This event is generated when the audio volume changes.
15	**waiting** This event is generated when the requested operation (such as playback) is delayed pending the completion of another operation (such as a seek).

Following is the example which allows to play the given video –

```
<!DOCTYPE HTML>

<html>
  <head>

    <script type = "text/javascript">
      function PlayVideo() {
        var v = document.getElementsByTagName("video")[0];
```

```
            v.play();
         }
      </script>
   </head>

   <body>

      <form>
         <video width = "300" height = "200" src = "/html5/foo.mp4">
         Your browser does not support the video element.
         </video>
         <br />
         <input type = "button" onclick = "PlayVideo();" value = "Play"/>
      </form>

   </body>
</html>
```