HTML5 and CSS3

HTML5: What is it?

HTML5 will be the new standard for HTML, XHTML, and the HTML DOM (document object model).

The previous version of HTML came in 1999. The web has changed a lot since then.

HTML5 is still a work in progress, but most modern browsers have some HTML5 support.

HTML5: Origins

- HTML5 is a cooperation between the World Wide Web Consortium (W3C) and the Web Hypertext Application Technology Working Group (WHATWG).
- WHATWG : Web Hypertext Application Technology Working Group
- WHATWG was working with web forms and applications, and W3C was working with XHTML 2.0. In 2006, they decided to cooperate and create a new version of HTML.

HTML5: Ground Rules

Some rules for HTML5 were established:

> New features should be based on HTML, CSS, DOM, and JavaScript
> Reduce the need for external plugins
> Better error handling
> More markup to replace scripting
> HTML5 should be device independent
> Dev process should be visible to the public

HTML5: New Features

- Canvas element for drawing
- Video/audio elements for media playback
- Better support for local offline storage
- New content specific elements, like article, footer, header, nav, section
- New form controls, like calendar, date, time,email, url, search

HTML5: Support

HTML5 is not yet an official standard, and no browser has full HTML5 support.

Darn it.

HTML5: Support

You may well ask: “How can I start using HTML5 if older browsers don’t support it?” But the question itself is misleading.

HTML5 is not one big thing; it is a collection of individual features. You can only detect support for individual features, like canvas, video, or geolocation.

HTML5: Support

Love it or hate it, you can’t deny that HTML 4 is the most successful markup format ever. HTML5 builds on that success.

You don’t need to throw away your existing markup. You don’t need to relearn things you already know. If your web application worked yesterday in HTML 4, it will still work today in HTML5. Period.

HTML5: Example

HTML5 supports all the form controls from HTML 4, but it also includes new input controls. Some of these are long-overdue additions like sliders and date pickers; others are more subtle...

HTML5: Example

For example, the email input type looks just like a text box, but mobile browsers will customize their onscreen keyboard to make it easier to type email addresses. Older browsers that don’t support the email input type will treat it as a regular text field, and the form still works with no markup changes or scripting hacks.

HTML5: DOCTYPE

The DOCTYPE which comes before the beginning <html> tag is much simpler in HTML 5. Here are some examples of what it looks like now...

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

HTML: DOCTYPE

Previous versions of HTML defined a lot of doctypes, and choosing the right one could be tricky. In HTML5, there is only one doctype:

```
<!DOCTYPE html lang=“en”>
```

HTML: Support

Whether you want to draw on a canvas, play video, design better forms, or build web applications that work offline, you'll find that HTML5 is already well-supported.

HTML: Support

Firefox, Safari, Chrome, Opera, and mobile browsers already support canvas, video, geolocation, local storage, and more. Google already supports microdata annotations. Even Microsoft — rarely known for blazing the trail of standards support — will be supporting most HTML5 features in the upcoming Internet Explorer 9.

HTML5: MIME types

Every time your web browser requests a page, the web server sends “headers” before it sends the actual page markup. Headers are important, because they tell your browser how to interpret the page markup that follows.

MIME = Multipurpose Internet Mail Extensions

HTML5: MIME types

The most important header is called Content-Type, and it looks like this:

```
Content-Type: text/html
```

“text/html” is called the “content type” or “MIME type” of the page.

HTML5: MIME types

This header is the only thing that determines what a particular resource truly is, and therefore how it should be rendered. Images have their own MIME types (image/jpeg for JPEG images, image/png for PNG images, and so on). JavaScript files have their own MIME type. CSS stylesheets have their own MIME type. Everything has its own MIME type. The web runs on MIME types.

HTML5: Detection

When your browser renders a web page, it constructs a Document Object Model, a collection of objects that represent the HTML elements on the page. Every element is represented in the DOM by a different object.

In browsers that support HTML5 features, certain objects will have unique properties. A quick peek at the DOM will tell you which features are supported.

HTML5: Detection

[Modernizr] is an open source, MIT-licensed JavaScript library that detects support for many HTML5 & CSS3 features. To use it, include the following <script> element at the top of your page…

HTML5: Detection

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>HTML5 sure is fun</title>
  <script src="modernizr.min.js"></script>
</head>
<body>
  ...
</body>
</html>
```

HTML5: Detection

Modernizr runs automatically. When it runs, it creates a global object called Modernizr, that contains a set of Boolean properties for each feature it can detect. For example, if your browser supports the canvas API, the Modernizr.canvas property will be true – otherwise the property will be false.

HTML5: Detection

```
if (Modernizr.canvas) {
  // let's draw some shapes!
} else {
  // no native canvas support available :(
}
```

More on HTML5 detection (and Modernizr) here: http://diveintohtml5.org/detect.html

HTML5: What's New

The internet has changed a lot since HTML 4.01 became a standard in 1999.

Today, some elements in HTML 4.01 are obsolete, never used, or not used the way they were intended to be. These elements are deleted or re-written in HTML5.

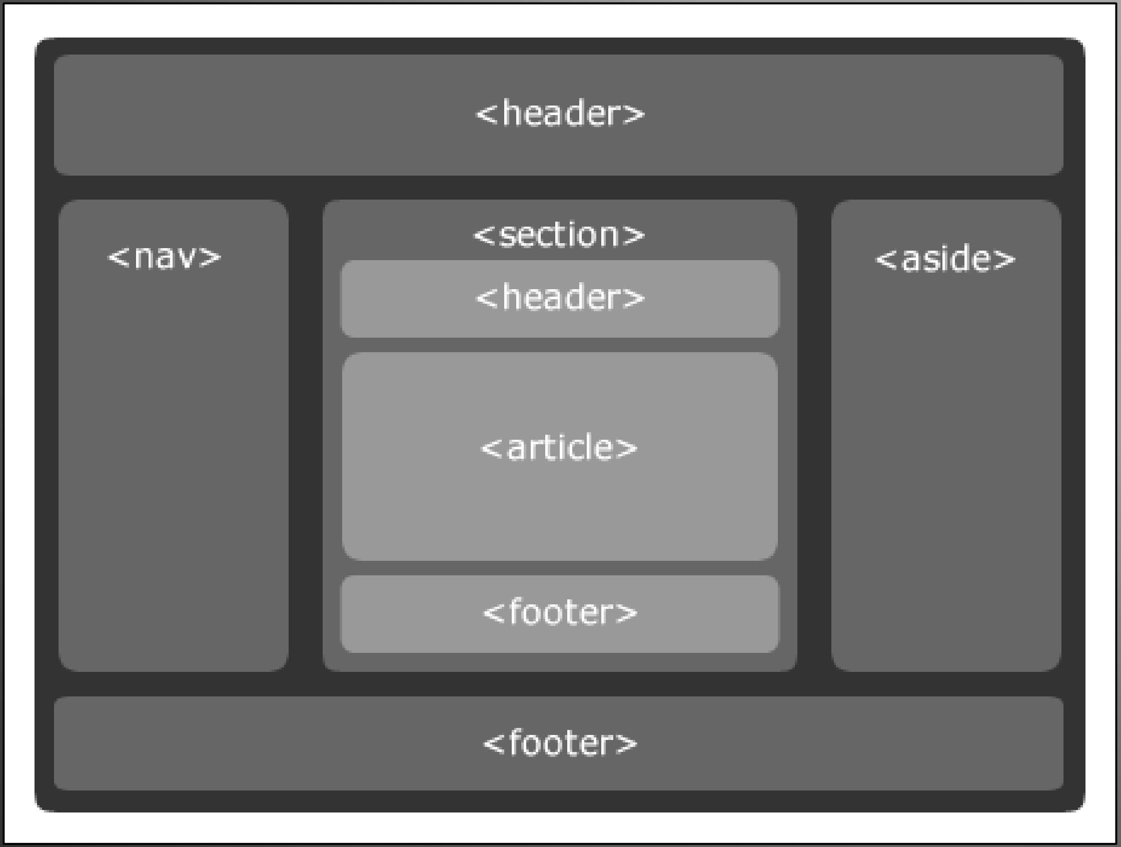
HTML5 also includes new elements for better structure, drawing, media content, form handling.

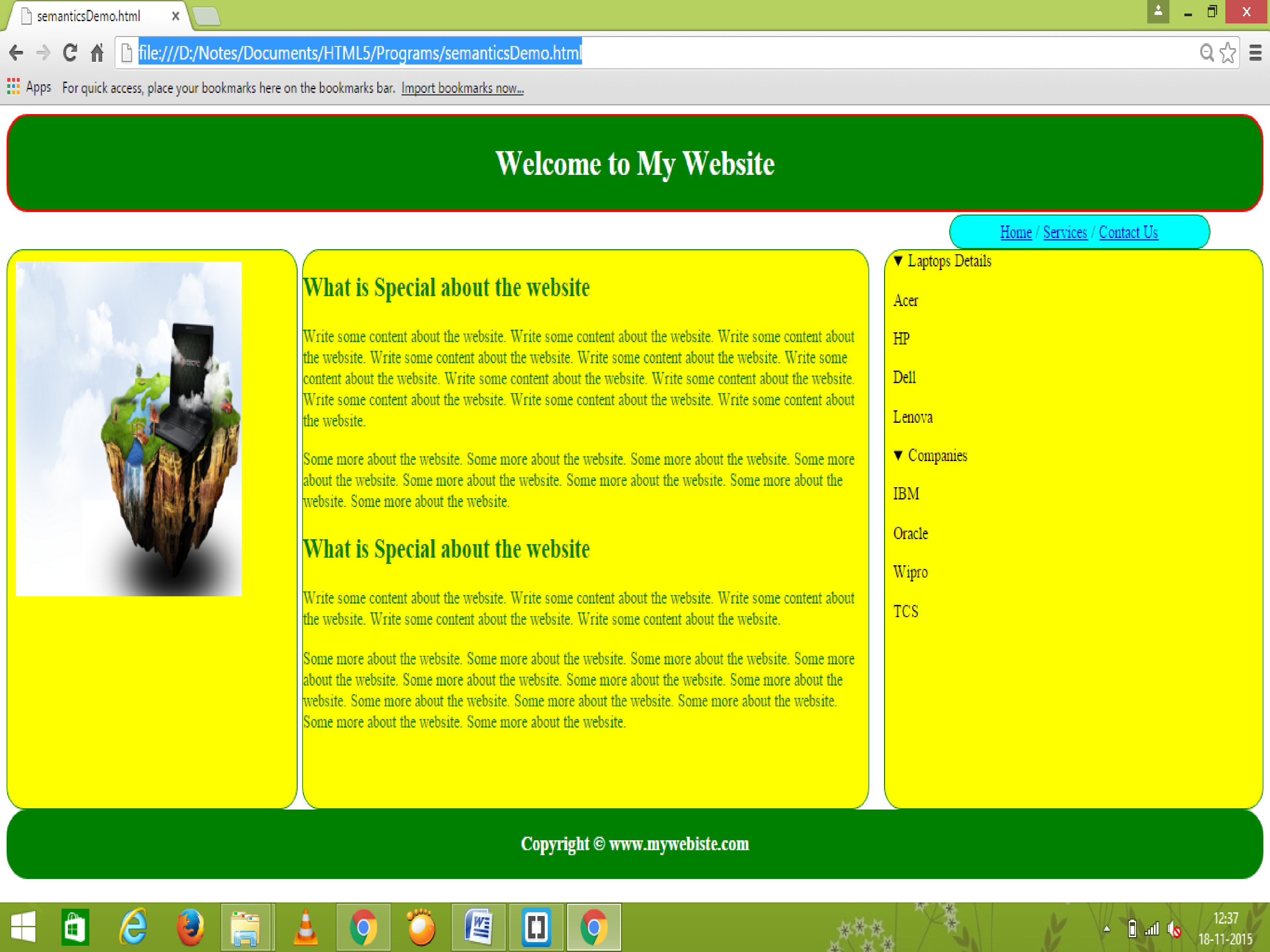
HTML5: What's New

New elements list:
http://www.w3schools.com/html5/

Some of the new markup elements are handy for creating better page structure. See the next slide for a representation of some of these elements.

HTML5: What's New




semanticsDemo.html
file:///D:/Notes/Documents/HTML5/Programs/semanticsDemo.html
Apps
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...

Welcome to My Website

Home / Services / Contact Us

What is Special about the website

Write some content about the website. Write some content about the website. Write some content about the website. Write some content about the website. Write some content about the website. Write some content about the website. Write some content about the website. Write some content about the website. Write some content about the website. Write some content about the website. Write some content about the website.

Some more about the website. Some more about the website. Some more about the website. Some more about the website. Some more about the website. Some more about the website. Some more about the website. Some more about the website.

What is Special about the website

Write some content about the website. Write some content about the website. Write some content about the website. Write some content about the website. Write some content about the website.

Some more about the website. Some more about the website. Some more about the website. Some more about the website. Some more about the website. Some more about the website. Some more about the website. Some more about the website. Some more about the website. Some more about the website. Some more about the website.

▼ Laptops Details

Acer

HP

Dell

Lenova

▼ Companies

IBM

Oracle

Wipro

TCS

Copyright © www.mywebiste.com

CSS Properties

- background-color: green;
- padding: 2px;
- color: white;
- text-align: center;
- border: 3px solid red;
- border-radius: 25px;
- margin-left: 10px;
- float : left;

HTML5: Video

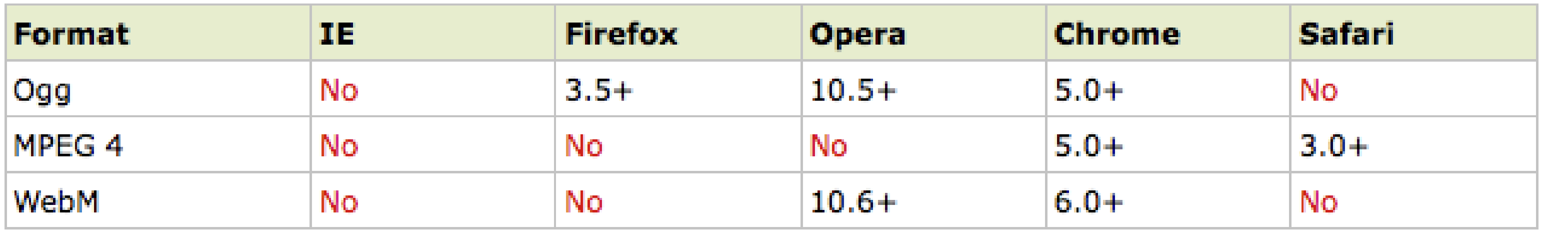
Until now, there hasn't been a standard for showing video on a web page.

Today, most videos are shown through a plugin (like Flash). However, not all browsers have the same plugins.

HTML5 specifies a standard way to include video with the video element.

HTML5: Video

Currently, there are 3 supported video formats for the video element:



Format	IE	Firefox	Opera	Chrome	Safari
Ogg	No	3.5+	10.5+	5.0+	No
MPEG 4	No	No	No	5.0+	3.0+
WebM	No	No	10.6+	6.0+	No

HTML5: Video

```
<!DOCTYPE HTML>
<html>
<body>

<video src="movie.ogg"  width="320"
height="240" controls="controls">
Your browser does not support the video tag.
</video>

</body>
</html>
```

HTML5: Video

The last example uses an Ogg file, and will work in Firefox, Opera and Chrome.

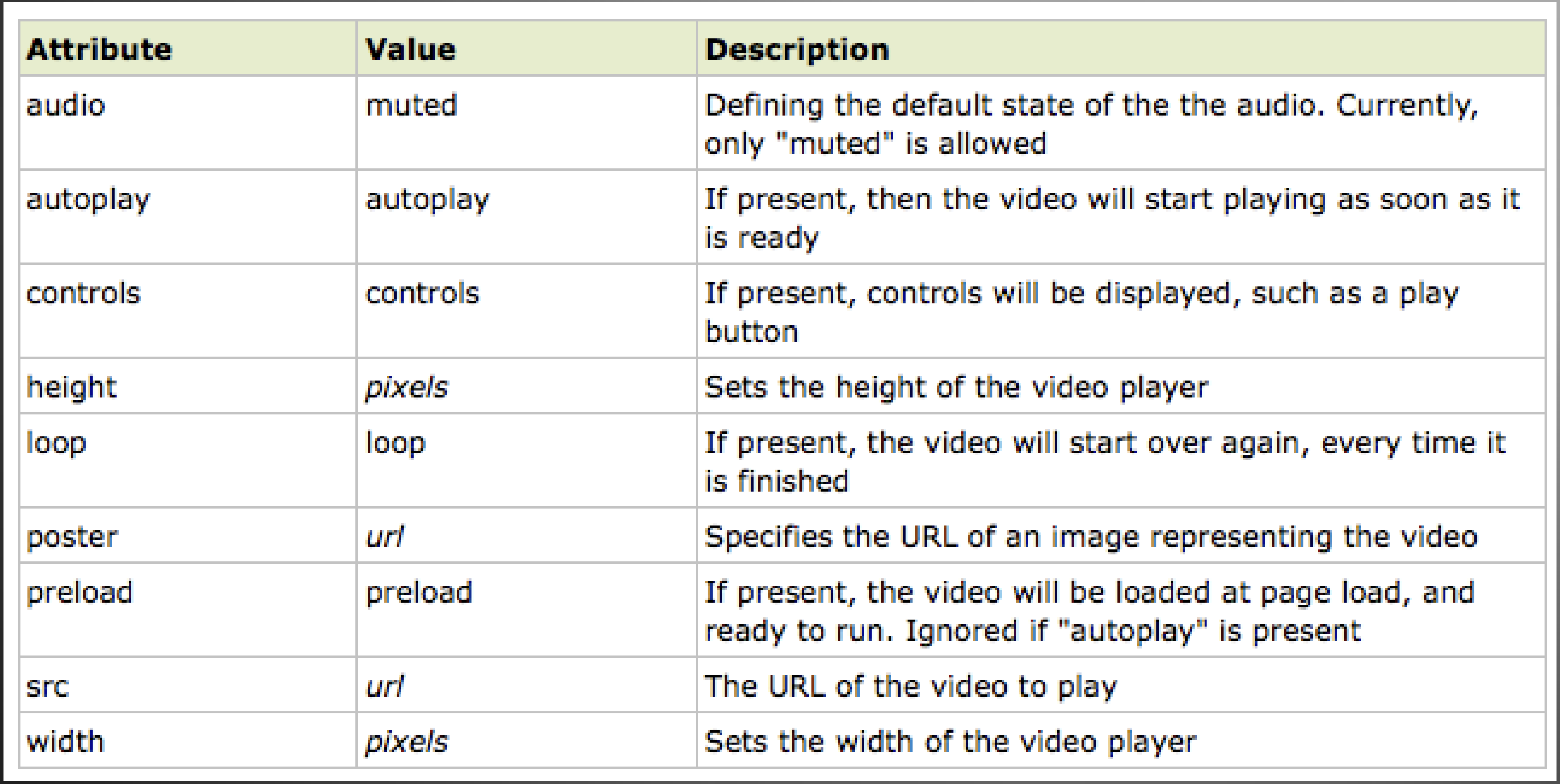
To make the video work in Safari and future versions of Chrome, we must add an MPEG4 and WebM file.

The video element allows multiple source elements. Source elements can link to different video files. The browser will use the first recognized format:

HTML5: Video

```
<video width="320" height="240"
controls="controls">
  <source src="movie.ogg" type="video/ogg" />
  <source src="movie.mp4" type="video/mp4"
/>
  <source src="movie.webm"
type="video/webm" />
Your browser does not support the video tag.
</video>
```

HTML5: Video



Attribute	Value	Description
audio	muted	Defining the default state of the the audio. Currently, only "muted" is allowed
autoplay	autoplay	If present, then the video will start playing as soon as it is ready
controls	controls	If present, controls will be displayed, such as a play button
height	*pixels*	Sets the height of the video player
loop	loop	If present, the video will start over again, every time it is finished
poster	*url*	Specifies the URL of an image representing the video
preload	preload	If present, the video will be loaded at page load, and ready to run. Ignored if "autoplay" is present
src	*url*	The URL of the video to play
width	*pixels*	Sets the width of the video player

HTML5: Audio

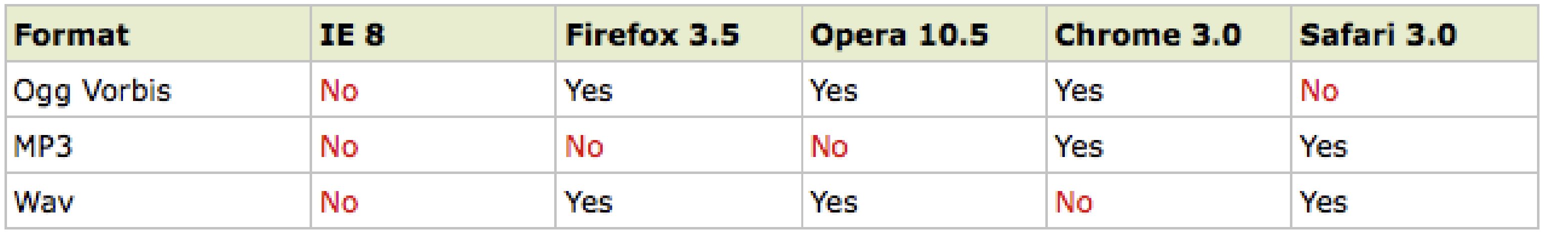
Until now, there has never been a standard for playing audio on a web page.

Today, most audio is played through a plugin (like Flash). However, not all browsers have the same plugins.

HTML5 specifies a standard way to include audio, with the audio element. The audio element can play sound files, or an audio stream.

HTML5: Audio

Currently, there are 3 supported formats for the audio element:



Format	IE 8	Firefox 3.5	Opera 10.5	Chrome 3.0	Safari 3.0
Ogg Vorbis	No	Yes	Yes	Yes	No
MP3	No	No	No	Yes	Yes
Wav	No	Yes	Yes	No	Yes

HTML5: Audio

```
<!DOCTYPE HTML>
<html>
<body>

<audio src="song.ogg" controls="controls">
Your browser does not support the audio
element.
</audio>

</body>
</html>
```

HTML5: Audio

The last example uses an Ogg file, and will work in Firefox, Opera and Chrome.

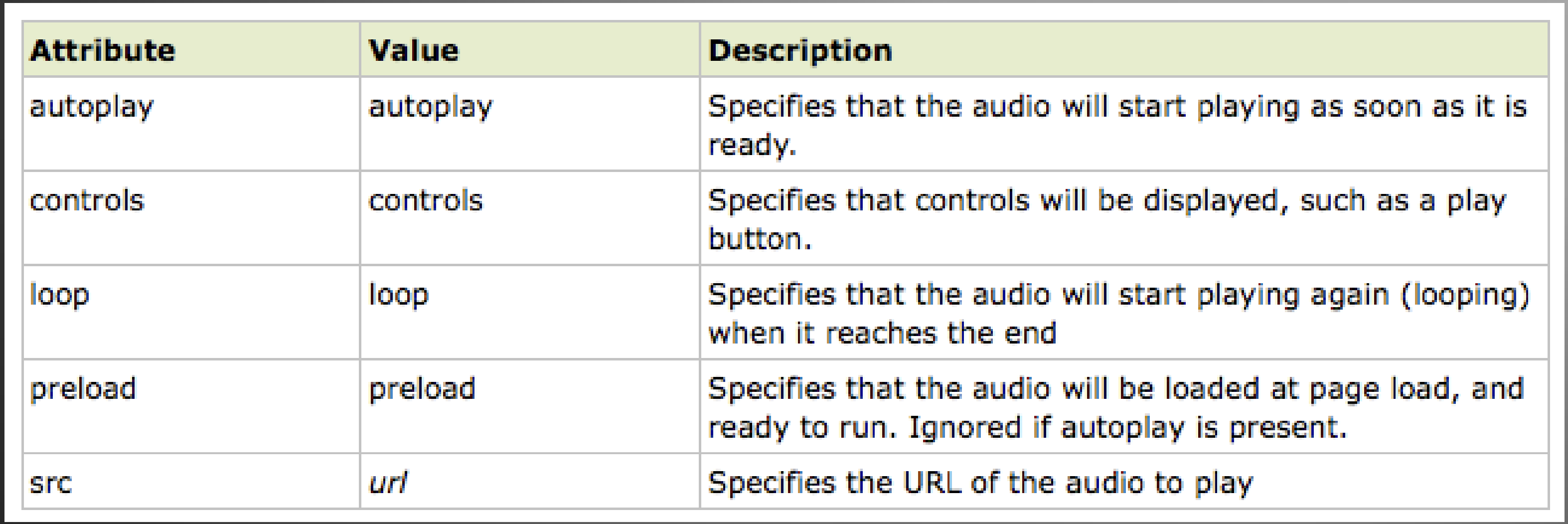
To make the audio work in Safari, the audio file must be of type MP3 or Wav.

The audio element allows multiple source elements. Source elements can link to different audio files. The browser will use the first recognized format.

HTML5: Audio

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

HTML5: Audio



Attribute	Value	Description
autoplay	autoplay	Specifies that the audio will start playing as soon as it is ready.
controls	controls	Specifies that controls will be displayed, such as a play button.
loop	loop	Specifies that the audio will start playing again (looping) when it reaches the end
preload	preload	Specifies that the audio will be loaded at page load, and ready to run. Ignored if autoplay is present.
src	*url*	Specifies the URL of the audio to play

HTML5: Canvas

The HTML5 canvas element uses JavaScript to draw graphics on a web page.

A canvas is a rectangular area, and you control every pixel of it.

The canvas element has several methods for drawing paths, boxes, circles, characters, and adding images.

HTML5: Canvas

Adding a canvas element to the HTML5 page.

Specify the id, width, height of the element:

```
<canvas id="myCanvas" width="200"
height="100"></canvas>
```

HTML5: Canvas

The canvas element has no drawing abilities of its own. All drawing must be done inside a JavaScript:

```
<script type="text/javascript">
var c=document.getElementById("myCanvas");
var cxt=c.getContext("2d");
cxt.fillStyle="#FF0000";
cxt.fillRect(0,0,150,75);
</script>
```

Draw text content

```
var myCan =
document.getElementById("obj1").getContext(
"2d");
myCan.font='20pt Arial';
myCan.fillStyle='red';
myCan.fillText('Welcome to HTML5
Context',150,100);
```

Draw a line

```
var myCan =
document.getElementById("obj1").getContext(
"2d");
myCan.beginPath();
myCan.moveTo(100,100);
myCan.lineTo(450,);
myCan.stroke();
```

Draw arc

```
var canvas = document.getElementById('obj1');
var context = canvas.getContext('2d');
var x = canvas.width / 2;
var y = canvas.height / 2;
var radius = 75;
var startAngle = 1.1 * Math.PI;
var endAngle = 1.9 * Math.PI;
var counterClockwise = false;

context.beginPath();
context.arc(x, y, radius, startAngle, endAngle, counterClockwise);
context.lineWidth = 15;
   // line color
context.strokeStyle = 'orange';
context.stroke();

```

Draw circle

- var canvas = document.getElementById('obj1');
- var context = canvas.getContext('2d');
- var centerX = canvas.width / 2;
- var centerY = canvas.height / 2;
- var radius = 70;
- context.beginPath();
- context.arc(centerX, centerY, radius, 0, 2 * Math.PI, false);
- context.fillStyle = '#abc123';
- context.fill();
- context.lineWidth = 5;
- context.strokeStyle = 'red';
- context.stroke();

Draw rectangle

```
var canvas = document.getElementById('obj1');
      var context = canvas.getContext('2d');

      context.beginPath();
      context.rect(188, 50, 200, 100);
      context.fillStyle = 'yellow';
      context.fill();
      context.lineWidth = 7;
      context.strokeStyle = 'red';
      context.stroke();
```

SVG

- SVG stands for Scalable Vector Graphics.
- SVG defines vector-based graphics in XML format.
- SVG defines the graphics in XML format
- SVG graphics do NOT lose any quality if they are zoomed or resized
- SVG integrates with other W3C standards such as the DOM and XSL

Draw text

```
<svg height=“60" width=“60">
 <text x="0" y="15" fill="red">
Welcome to SVG Text Example
</text>
</svg>
```

Draw line

```
<svg height=“100" width=“400">
 <line x1="0" y1="0" x2="200" y2="200"
   style="stroke:rgb(255,0,0);
    stroke-width:2" />
</svg>
```

Draw a circle

```
<svg height="100" width="100">
  <circle cx="50" cy="50" r="40"
  stroke="black" stroke
width="3" fill="red" />
</svg>
```

Draw eclipse

-
  ```
  <svg width="400" height="110">
    <rect width="300" height="100"
    style="fill:rgb(0,0,255);stroke-
    width:3;stroke:rgb(0,0,0)" />
    </svg>
  ```

Some important pre-defined shape elements

- Rectangle <rect>
- Circle <circle>
- Ellipse <ellipse>
- Line <line>
- Polyline <polyline>
- Polygon <polygon>
- Path <path>

HTML5: Web Storage

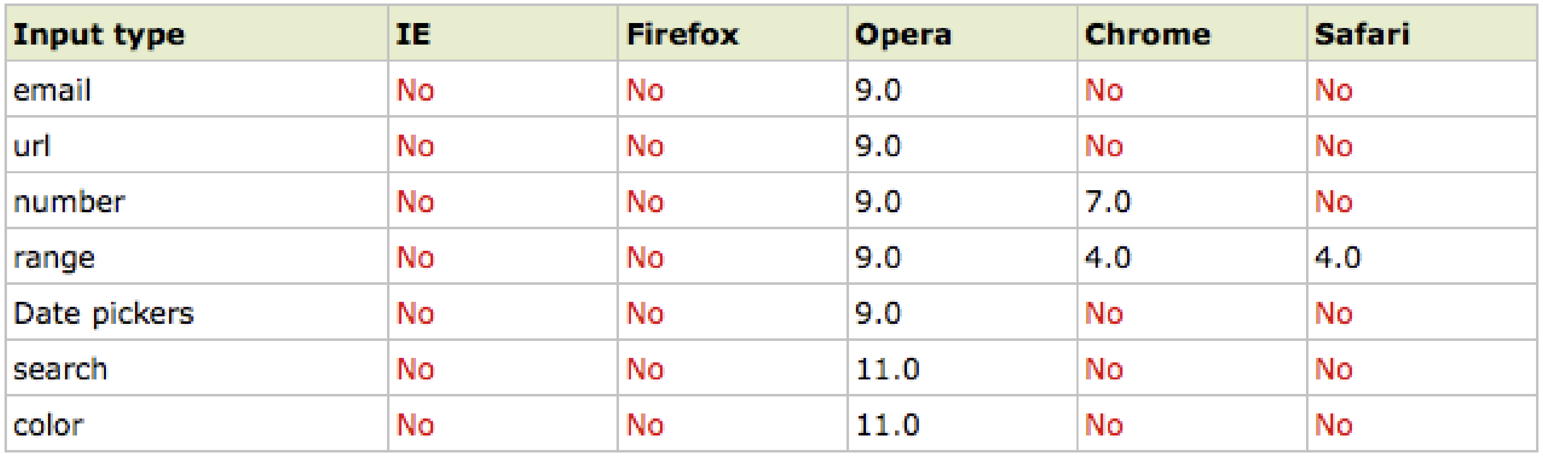
For information about web storage:
http://www.w3schools.com/html5/

HTML5: Input types

HTML5 has several new input types for forms.

- email
- url
- number
- range
- date pickers (date, month, week, time, datetime, datetime-local)
- search
- color

HTML5: Input types



Input type	IE	Firefox	Opera	Chrome	Safari
email	No	No	9.0	No	No
url	No	No	9.0	No	No
number	No	No	9.0	7.0	No
range	No	No	9.0	4.0	4.0
Date pickers	No	No	9.0	No	No
search	No	No	11.0	No	No
color	No	No	11.0	No	No

Note: Opera has the best support for the new input types. However, you can already start using them in all major browsers. If they are not supported, they will behave as regular text fields.

HTML5: Input - e-mail

The email type is used for input fields that should contain an e-mail address.

The value of the email field is automatically validated when the form is submitted.

E-mail: <input type="email" name="user_email" />

Tip: Safari on the iPhone recognizes the email input type, and changes the on-screen keyboard to match it (adds @ and .com options).

HTML5: Input – url

The url type is used for input fields that should contain a URL address.

The value of the url field is automatically validated when the form is submitted.

Homepage: <input type="url" name="user_url" />

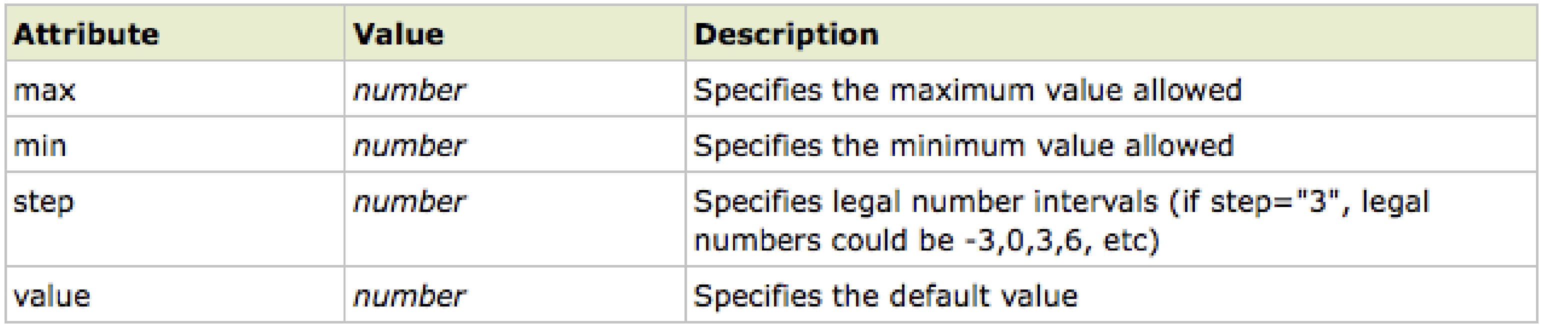
Tip: Safari on the iPhone recognizes the url input type, and changes the on-screen keyboard to match it (adds .com option).

HTML5: Input – number

The number type is used for input fields that should contain a numeric value.

Set restrictions on what numbers are accepted:

```
Points: <input type="number" name="points"
min="1" max="10" />
```



Attribute	Value	Description
max	*number*	Specifies the maximum value allowed
min	*number*	Specifies the minimum value allowed
step	*number*	Specifies legal number intervals (if step="3", legal numbers could be -3,0,3,6, etc)
value	*number*	Specifies the default value

HTML5: Input – range

The range type is used for input fields that should contain a value from a range of numbers.

The range type is displayed as a slider bar.

You can also set restrictions on what numbers are accepted:

```
<input type="range" name="points" min="1"
max="10" />
```

HTML5: Input – date pickers

HTML5 has several new input types for selecting date and time:

- date – Selects date, month and year
- month – Selects month and year
- week – Selects week and year
- time – Selects time (hour and minute)
- datetime – Selects time, date, month and year
- datetime-local – Selects time, date, month and year (local time)

HTML5: Input – search

The search type is used for search fields like a site search or Google search.

The search field behaves like a regular text field.

HTML5: Input – color picker

The color type is used for input fields that should contain a color.

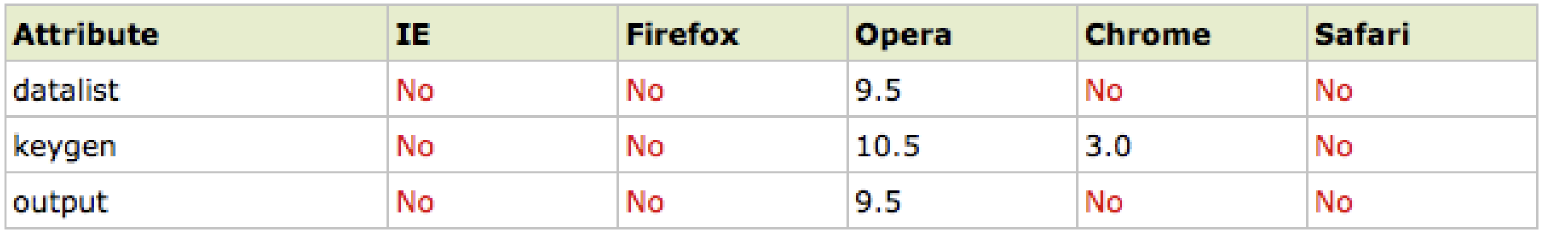
This input type will allow you to select a color from a color picker:

Color: <input type="color" name="user_color" />

HTML5: Input – form

HTML5 has several new elements and attributes for forms.

- datalist
- keygen
- output



Attribute	IE	Firefox	Opera	Chrome	Safari
datalist	No	No	9.5	No	No
keygen	No	No	10.5	3.0	No
output	No	No	9.5	No	No

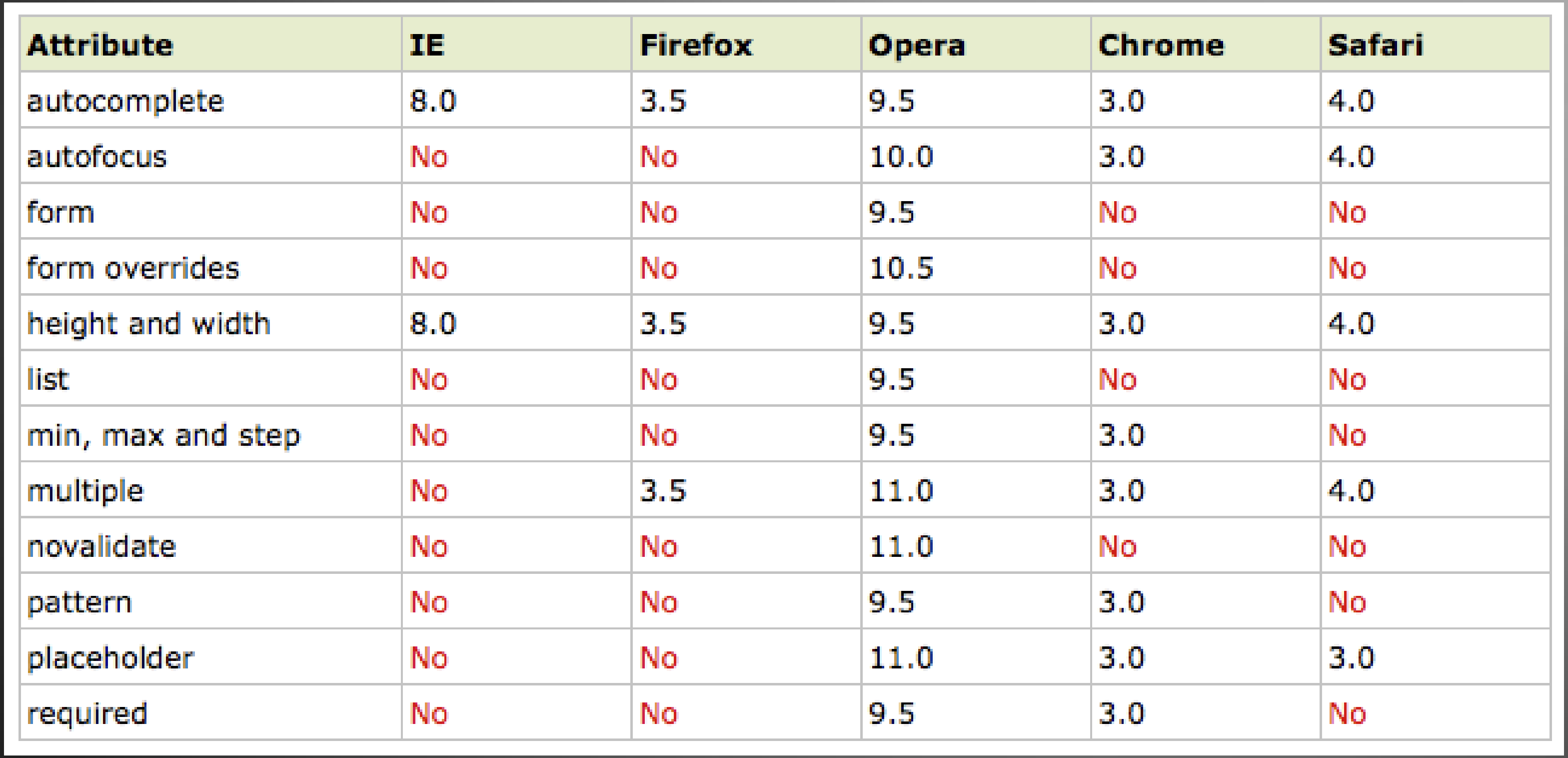
HTML5: Form elements

The datalist element specifies a list of options for an input field. The list is created with option elements inside the datalist.

The purpose of the keygen element is to provide a secure way to authenticate users.

The output element is used for different types of output, like calculations or script output:

HTML5: Form attributes



Attribute	IE	Firefox	Opera	Chrome	Safari
autocomplete	8.0	3.5	9.5	3.0	4.0
autofocus	No	No	10.0	3.0	4.0
form	No	No	9.5	No	No
form overrides	No	No	10.5	No	No
height and width	8.0	3.5	9.5	3.0	4.0
list	No	No	9.5	No	No
min, max and step	No	No	9.5	3.0	No
multiple	No	3.5	11.0	3.0	4.0
novalidate	No	No	11.0	No	No
pattern	No	No	9.5	3.0	No
placeholder	No	No	11.0	3.0	3.0
required	No	No	9.5	3.0	No

HTML5 drag and drop

- From HTML5 it is possible to drag and drop HTML elements inside an HTML page.
- Via JavaScript event listeners you can decide what happens when the user drags and drops elements.
- During drag and drop HTML elements can take on two roles.
  - Draggable
  - Drop target/drop zone

Continue...

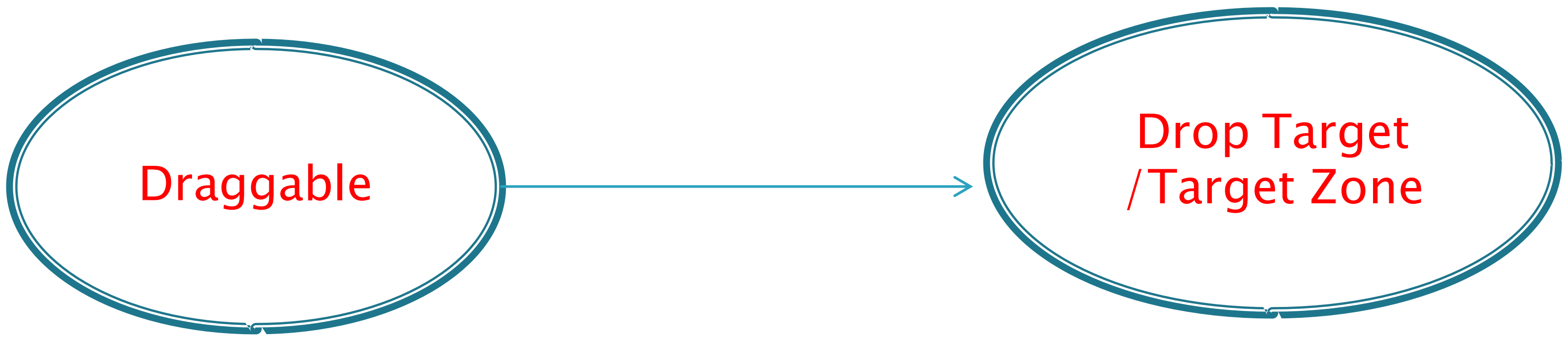
- Draggable: The draggable element is the element which the user should be able to drag around the HTML page and drop somewhere, for some action to occur. This can be one or more elements.
- Drop target/drop zone: The drop target or drop zone is the element onto which the draggable elements are dropped.

Pts to achieve drag and drop

- Set the HTML elements to draggable.
- Attach event listeners to the draggable HTML elements.
- Attach event listeners to the drop target HTML elements.

Attribute:

draggable = true

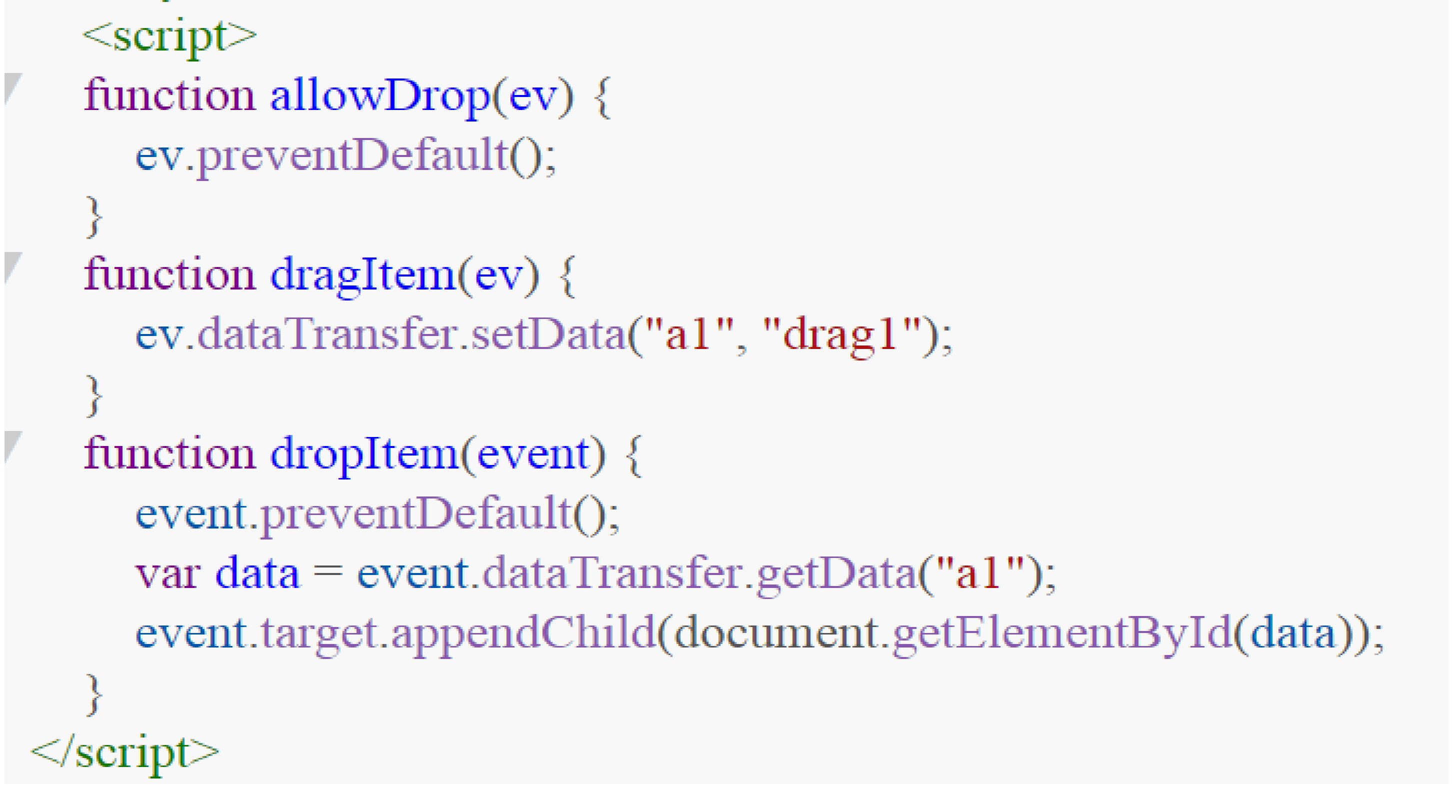


Events :

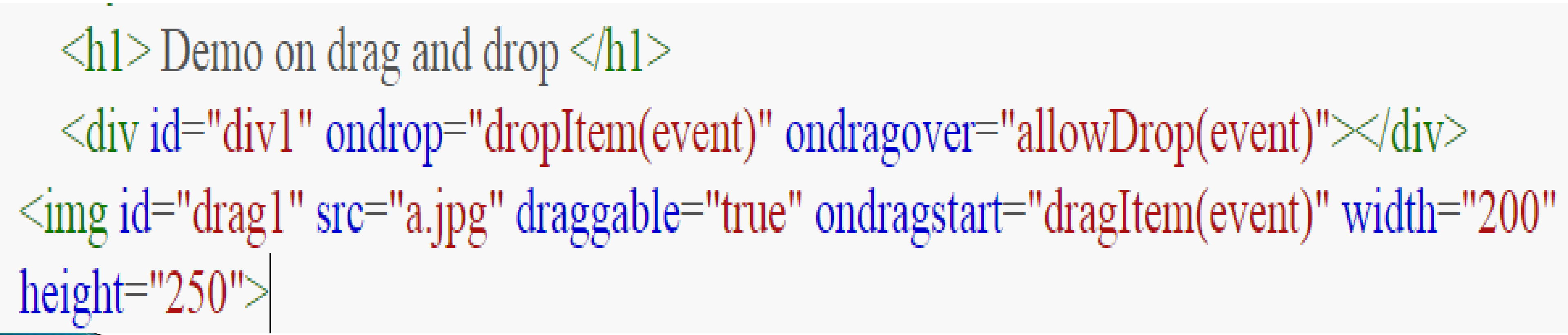
dragstart
dragend

Events :

dragenter
dragover
dragleave
drop



```
    <script>
    function allowDrop(ev) {
        ev.preventDefault();
    }
    function dragItem(ev) {
        ev.dataTransfer.setData("a1", "drag1");
    }
    function dropItem(event) {
        event.preventDefault();
        var data = event.dataTransfer.getData("a1");
        event.target.appendChild(document.getElementById(data));
    }
</script>
```



```
    <h1> Demo on drag and drop </h1>
    <div id="div1" ondrop="dropItem(event)" ondragover="allowDrop(event)"></div>
<img id="drag1" src="a.jpg" draggable="true" ondragstart="dragItem(event)" width="200"
height="250">
```

HTML5 Web Workers

- When executing scripts in an HTML page, the page becomes unresponsive until the script is finished.
- A web worker is a JavaScript that runs in the background, independently of other scripts, without affecting the performance of the page. You can continue to do whatever you want. clicking, selecting things etc. While the web worker runs in the background.

```
if(typeof(Worker) !="undefined") {
        alert("Yes it will support ")
}else {
        alert("No it won't support")
}
```

Sample external background script file

demo_worker.js

```
var i = 0;
timeCount();
function timeCount() {
     i = i+1;
     postMessage(i);
     setTimeOut("timeCount()",1000);
}
timeCount();
```

Send and receive msg from Worker

- var w = new Worker();
  w.onmessage = function(event)
  document.getElementById("result").innerHTML
  =event.data

}

- Terminate worker
  - w.terminate()

```
<p>Count numbers: <output id="result"></output></p>
<button onclick="startWorker()">Start Worker</button>
<button onclick="stopWorker()">Stop Worker</button>
<br><br>
<script>
var w;
function startWorker() {
   if(typeof(Worker) !== "undefined") {
      if(typeof(w) == "undefined") {
         w = new Worker("demo_worker.js");
      }
      w.onmessage = function(event) {
         document.getElementById("result").innerHTML = event.data;
      };
   } else {
      document.getElementById("result").innerHTML = "Sorry, your browser
does not support Web Workers...";
   }
}
function stopWorker() {
   w.terminate();
   w = undefined;
}
</script>
```

HTML Local Storage Objects

- Storage: It is a object which help to store the data offline.
- HTML local storage provides two objects for storing data on the client.
  - localStorage
  - sessionStorage

Continue...

- localStorage: The localStorage object stores the data with no expiration date. The data will not be deleted when the browser is closed, and will be available the next day, week or year.
- The sessionStorage object is equal to the localStorage object, except that it stores the data for only one session. The data is deleted when the user closes the browser window.

Set the value and get the value

- Store
  - sessionStorage.variableName = value
  - localStorage.variableName = value
- Get
  - var num
  - If(sessionStorage.variableName){
  - num = eval(sessionStorage.variableName)
  - }

GeoLocation

- The HTML5 geolocation features lets you find out the geographic coordinates (latitude and logitude) of the current location of your website's visitor.
- Getting the position information of the site visitor using the HTML5 geolocation API is fairly simple.

Sample Example

```
if(navigator.geolocation){
navigator.geolocation.getCurrentPosition(
function(position){
var positionInfo="Latitude is
"+position.coords.latitude+" and Longitude is
"+position.coords.longitude;
                    });
      }
```