# HTML5 Presentation

Michelle McCausland

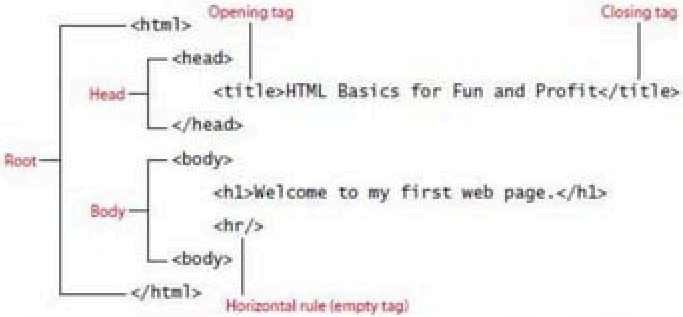
# Summary

1. What is HTML
2. History of HTML
3. What is HTML5
4. New elements in HTML5
5. New APIs
    - Geolocation
    - Drag & Drop
    - Local Storage
    - Application Cache
    - SSE

# 1. What is HTML

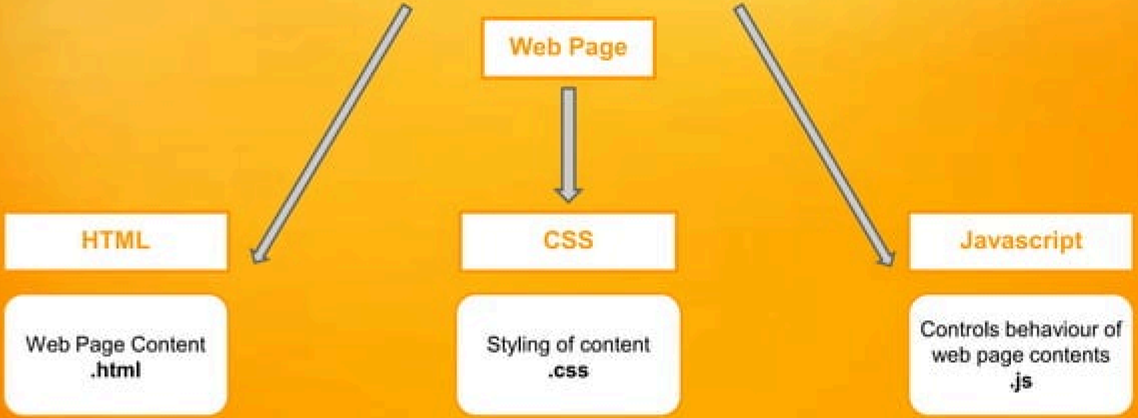
- It stands for **Hyper Text Markup Language**
- The building blocks of a web page
- All content is defined between **opening** and **closing tags**:

Basic HTML Code Structure



```
<html>
    <head>
        <title>HTML Basics for Fun and Profit</title>
    </head>
    <body>
        <h1>Welcome to my first web page.</h1>
        <hr/>
    <body>
</html>
```

HTML Cheat Sheet



Web Page

HTML

Web Page Content
**.html**

CSS

Styling of content
**.css**

Javascript

Controls behaviour of web page contents
**.js**

```
<html>
   <head>Page Title</head>
   <body>
      <h1>Hello World</h1>
      <p>This is some dummy
text!</p>
   </body>
</html>
```

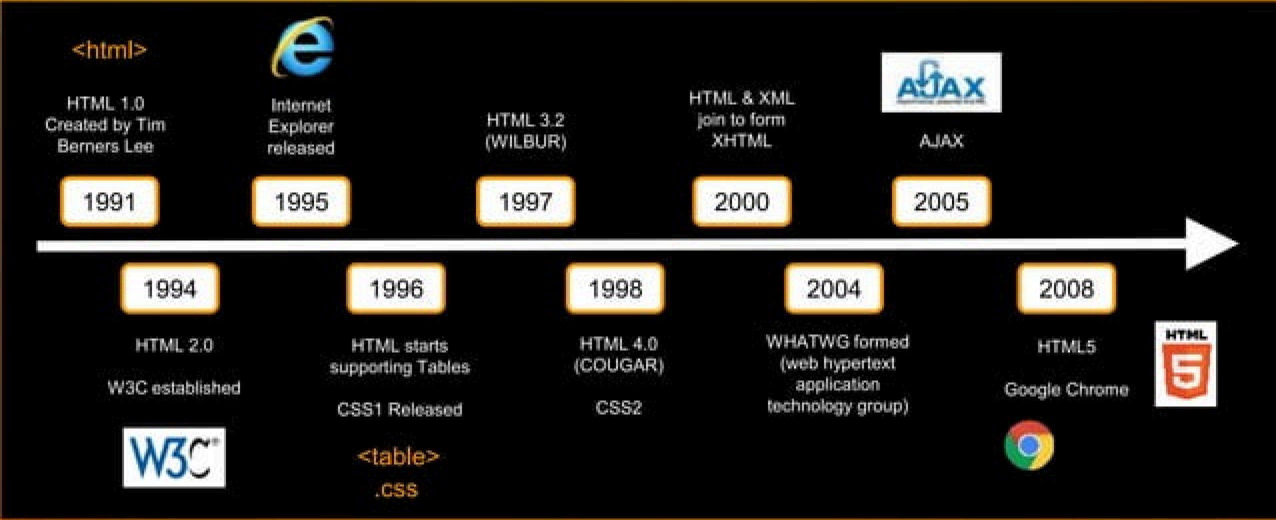
```
h1{text-align:center;
font-weight: bold; font-
size:16px;}

p{text-align:left; font-
size:10px;}
```

```
document.getElementById
("h1").innerHTML = "Hello
World";

console.log(5 + 6);
```

# 2. History of HTML



<html>

1991

HTML 1.0 Created by Tim Berners Lee

1994

HTML 2.0

W3C established

1995

Internet Explorer released

1996

HTML starts supporting Tables

CSS1 Released

<table>

.css

1997

HTML 3.2 (WILBUR)

1998

HTML 4.0 (COUGAR)

CSS2

2000

HTML & XML join to form XHTML

2004

WHATWG formed (web hypertext application technology group)

2005

AJAX

2008

HTML5

Google Chrome

# 3. What is HTML5

- The fifth revision of the HTML standard of the W3C
- It is the successor of **HTML 4.0** which was standardized in 1998
- Released in **2008**
- It is supported by all newer versions of major browsers, however older versions may not render some elements correctly.
- All browsers automatically handle unrecognized elements as inline elements.
- HTML5 features **new elements** not in its predecessors:
    - A new doctype
    - Semantic elements
    - Form controls/ elements
    - Graphic elements
    - Multimedia elements
    - New APIs

# New Doctype

- The Doctype declaration must be the **first thing** you include on your web page.
- It is basically a message to your web browser to tell it what version of HTML the page is written in.

- HTML 4.0 Doctype:

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

- HTML5 Doctype:

```
<!DOCTYPE html>
```

As you can see the HTML5 doctype is much shorter and easier to remember!

# Semantic Elements

Non-Semantic Elements - For example:<div>

Tells us nothing about the content being included

Semantic Elements - For example: <form> <table>

- Clearly defines the content.
- Easy for the browser to read
- Improves SEO (search engine optimization) for search engines
- Allows for better user experience

**HTML 4.0:**

```
<div id="header"> </div>

<div id="footer"> </div>
```

**HTML5:**

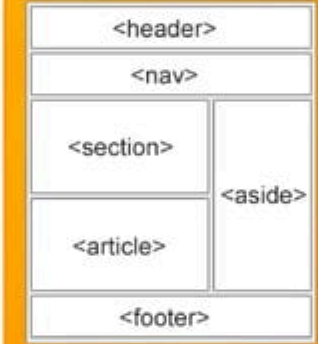
```
<header> </header>

<footer> </footer>
```

# Semantic Elements Ctd.

HTML5 contains many new semantic elements that make writing web pages much easier.

- **<header>** This section will typically contain the site name, a logo and or a navigation menu.
- **<nav>** This tag is used to identify a navigation bar containing navigation links.
- **<section>** This element defines a section in the web page.
- **<article>** This defines an article in the web page ie. an independent, self-contained piece of content. It is useful for blog posts, forum posts, newspaper articles etc.
- **<aside>** This tag can be used to create sidebars on a webpage.
- **<footer>** This tag is used at the bottom of the webpage. It normally includes author and copyright information as well as contact information and or navigation links



<header>

<nav>

<section>

<aside>

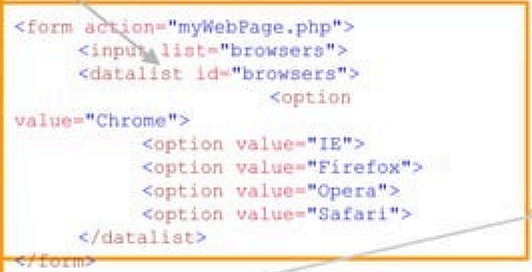
<article>

<footer>

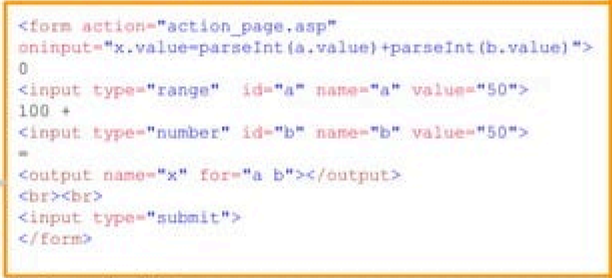
# Form Controls/ Elements

**New Form Elements:**

**<datalist>** This tag specifies a list of predefined options for an <input> element in a form.



```
<form action="myWebPage.php">
      <input list="browsers">
      <datalist id="browsers">
                        <option
value="Chrome">
            <option value="IE">
            <option value="Firefox">
            <option value="Opera">
            <option value="Safari">
      </datalist>
</form>
```



```
<form action="action_page.asp"
oninput="x.value=parseInt(a.value)+parseInt(b.value)">
0
<input type="range"  id="a" name="a" value="50">
100 +
<input type="number" id="b" name="b" value="50">
=
<output name="x" for="a b"></output>
<br><br>
<input type="submit">
</form>
```

**<output>** This tag is used to represent the result of a calculation

# Form Controls/ Elements Ctd.

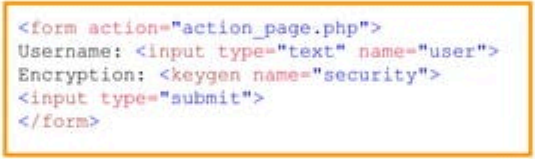
**<keygen>** This tag is used to provide user authentication securely.

When the form is submitted 2 keys are generated, 1 private & 1 public.

Private key is stored locally.

Public key is sent to server.

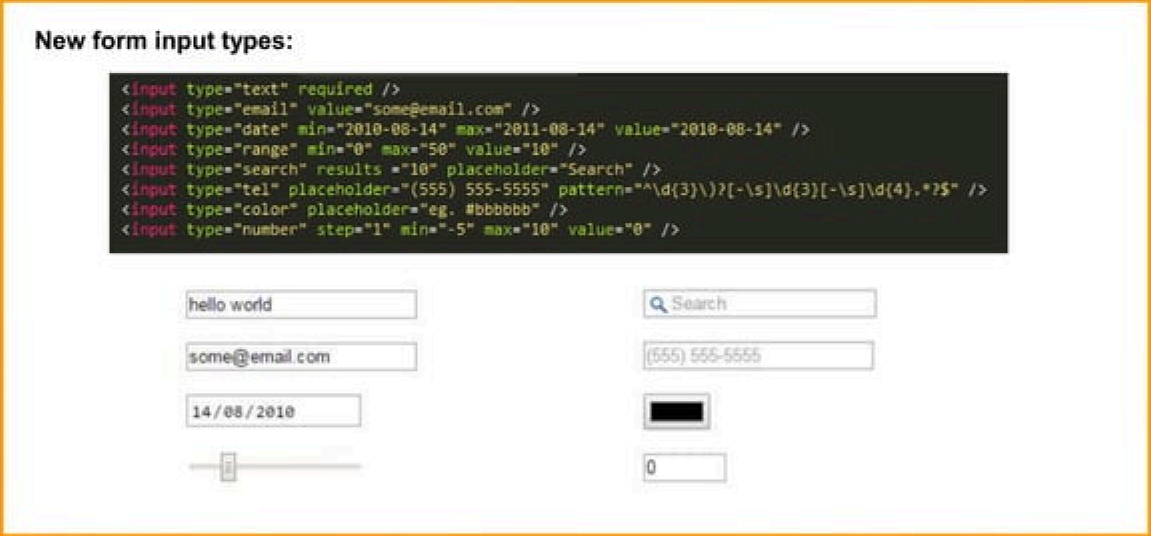
The public key could be used to generate a client certificate to authenticate user if needed.



```
<form action="action_page.php">
Username: <input type="text" name="user">
Encryption: <keygen name="security">
<input type="submit">
</form>
```

**Note:** By default, browsers don't display unknown elements.

# Form Controls/ Elements Ctd.



New form input types:

```
<input type="text" required />
<input type="email" value="some@email.com" />
<input type="date" min="2010-08-14" max="2011-08-14" value="2010-08-14" />
<input type="range" min="0" max="50" value="10" />
<input type="search" results ="10" placeholder="Search" />
<input type="tel" placeholder="(555) 555-5555" pattern="^\d{3}\)?[-\s]\d{3}[-\s]\d{4}.*?$" />
<input type="color" placeholder="eg. #bbbbbb" />
<input type="number" step="1" min="-5" max="10" value="0" />
```

hello world

some@email.com

14/08/2010

Search

(555) 555-5555

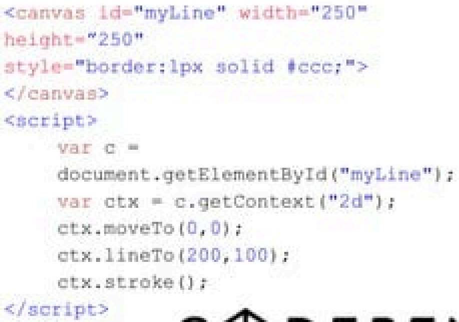
0

# Graphic Elements

## <canvas>

This tag is used as a container for canvas graphics.

***Basic Example - Line:***

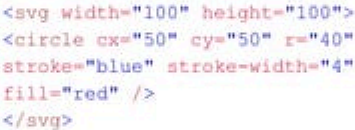


```
<canvas id="myLine" width="250"
height=”250"
style="border:1px solid #ccc;">
</canvas>
<script>
    var c =
    document.getElementById("myLine");
    var ctx = c.getContext("2d");
    ctx.moveTo(0,0);
    ctx.lineTo(200,100);
    ctx.stroke();
</script>
```

## <svg>

Stands for Scalable Vector Graphics

***Basic Example - Circle:***



```
<svg width="100" height="100">
<circle cx="50" cy="50" r="40"
stroke="blue" stroke-width="4"
fill="red" />
</svg>
```

# Graphic Elements Ctd.

**Differences between Canvas and SVG:**

## <canvas>

- It is very resolution dependent
- No support for event handlers
- Poor text rendering capabilities
- Good for graphic-intensive games
- Can save resulting image as .png .jpg

## <svg>

- Resolution independent
- Support for event handlers
- Slow rendering if complex (lots of DOM usage)
- Suitable for apps with large rendering areas eg. Google Maps
- Not suitable for game apps

# Multimedia Elements

Examples of multimedia on web pages include: images, videos, audio and more.

**Common image formats:** .jpg, .png, .gif

**Common video formats:** .avi, .mpg, .mov, .swf, .flv, .ogg, .mp4

**Common audio formats:** .mp3, .wav, .ogg, .aac, .midi

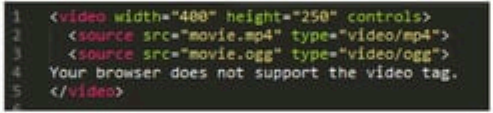
# Multimedia Elements Ctd.

**<video> element**

To embed a video onto your webpage you would include it within the video tag.

Before HTML5 there was no standard for including videos on a web page.

IE9+, Chrome, Firefox, Opera and Safari support this tag. However, IE8 and earlier does not.



```
<video width="400" height="250" controls>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
Your browser does not support the video tag.
</video>
```

Currently the audio tag supports 3 formats:

mp4, webm, ogg

| Browser | MP4 | WebM | Ogg |
|---|---|---|---|
| Internet Explorer | Yes | No | No |
| Chrome | Yes | Yes | Yes |
| Safari | Yes | No | No |
| Firefox | Yes | Yes | Yes |
| Opera | Yes(from Opera 25) | Yes | Yes |

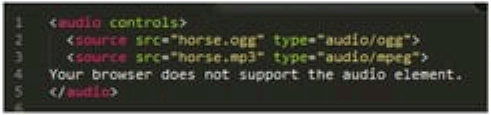
# Multimedia Elements Ctd.

**<audio> element**

If you wanted to embed a piece of audio on a webpage you include it within the audio tag.

Before HTML5 there was no standard for playing audio files on a web page. Audio files could only be played with a plugin such as Flash.

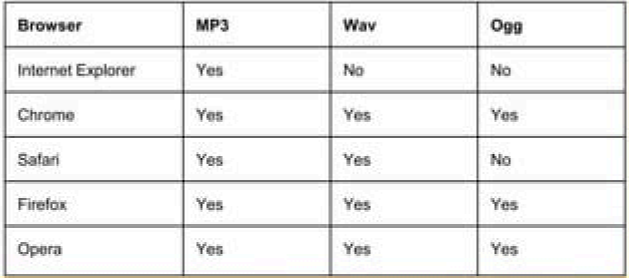
IE9+, Chrome, Firefox, Opera and Safari support this tag. However, IE8 and earlier does not.



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

Currently the audio tag supports 3 formats:

mp3, wav, ogg



| Browser | MP3 | Wav | Ogg |
|---|---|---|---|
| Internet Explorer | Yes | No | No |
| Chrome | Yes | Yes | Yes |
| Safari | Yes | Yes | No |
| Firefox | Yes | Yes | Yes |
| Opera | Yes | Yes | Yes |

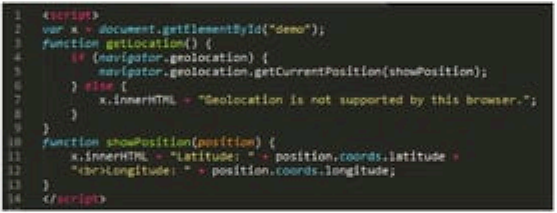
# New APIs

- API stands for **Application Program Interface**
- It can be described as a set of routines, protocols and tools for building software applications.
- A software company eg. Google can release their API (Google Maps) to the public so other developers can design software using its service.

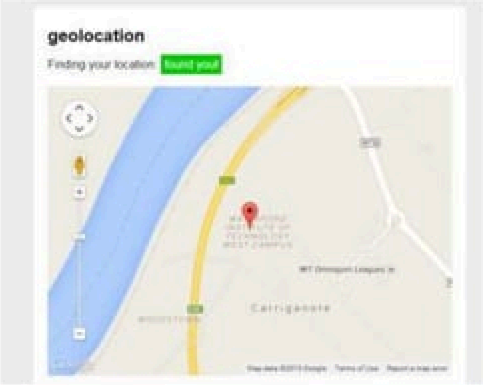
# Geolocation

The Geolocation API is used to determine location of a user, once approved by the user of course!

IE9+, Chrome, Firefox, Safari, Opera support Geolocation. However, it is much more accurate on devices with GPS like smart phones and tablets.



```
<script>
var x = document.getElementById("demo");
function getLocation() {
    if (navigator.geolocation) {
        navigator.geolocation.getCurrentPosition(showPosition);
    } else {
        x.innerHTML = "Geolocation is not supported by this browser.";
    }
}
function showPosition(position) {
    x.innerHTML = "Latitude: " + position.coords.latitude +
    "<br>Longitude: " + position.coords.longitude;
}
</script>
```



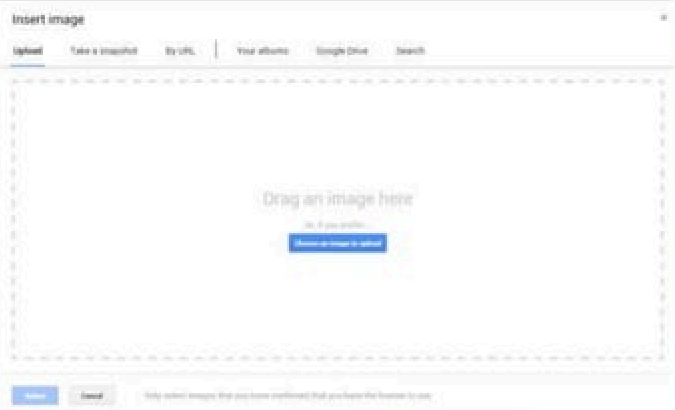
W3C Geolocation API Specification

CODEPEN

# Drag And Drop

- When you “grab” an object and drag it to a different location.
- In HTML5 any element can be draggable.
- IE9+, Firefox, Chrome, Opera,

Safari support drag & drop.



Drag an image here

Or, if you prefer...

Only select images that you have confirmed that you have the license to use.

CODEPEN

HTML5 Rocks Tutorial

# Local Storage

Web applications are allowed to store data locally within the user's browser.
Pre HTML5 this data was stored in cookies which were much less secure and were much smaller in size.

- Local storage has a storage limit of at least 5mb.
- It is secure as information is never transferred to the server.
- It does not affect the website's performance.
- Local storage is supported in IE8+, Chrome, Firefox, Opera, Safari. IE7 below does not support it.

HTML local storage provides 2 objects for storing data on the client:

1. window.localStorage - stores data with no expiration date
2. code.sessionStorage - stores data for one session (data is lost when tab is closed)

# Local Storage Ctd.

The localStorage object stores data with no expiration date ie. data will not be deleted when the browser is closed.

```
// Store
localStorage.setItem("lastname", "Smith");

// Retrieve
document.getElementById("result").innerHTML = localStorage.getItem("lastname");
```

**What's going on:**

- We create a localStorage name/value pair with name="lastname" and value="smith"
- Retrieve the value of “lastname” and insert it into the element with id=”result”
- localStorage.removeItem(“lastname”); >> removes lastname local storage item
- name/value pairs are always stored as strings

# Local Storage Ctd.

The following code counts the number of times a user has clicked a button in the current session.

```
if (sessionStorage.clickcount) {
    sessionStorage.clickcount = Number(sessionStorage.clickcount) + 1;
} else {
    sessionStorage.clickcount = 1;
}
document.getElementById("result").innerHTML = "You have clicked the button " +
sessionStorage.clickcount + " time(s) in this session.";
```

**sessionStorage** object is similar to the localStorage object however it stores the data for one session and the data is deleted when the user closes the browser window.

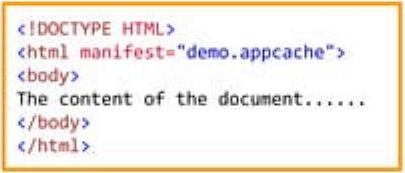
# Application Cache

This allows a web application to be accessible without an internet connection.

There are 3 advantages to this:

1. **Reduced server load** - The browser only downloads updated resources from the server.
2. **Speed** - Cached resources load faster.
3. **Offline Browsing** - Users can use the application offline

IE 10, Chrome, Firefox, Opera and Safari support application cache.



```
<!DOCTYPE HTML>
<html manifest="demo.appcache">
<body>
The content of the document......
</body>
</html>
```

Every page with the manifest attribute specified will be cached when the user visits it. The file extension for the manifest is **“.appcache”**

# Application Cache Ctd.

The manifest file is a simple text file which tells the browser what to cache.

It has 3 sections:

1. **Cache Manifest** - Files listed under this section will be cached after they are downloaded for the first time.
2. **Network**- These files require a connection to the server, and will never be cached.
3. **Fallback** - Files under this section specify fallback pages if a page is unaccessible.

Once an application is cached, it remains cached until one of the following happens:

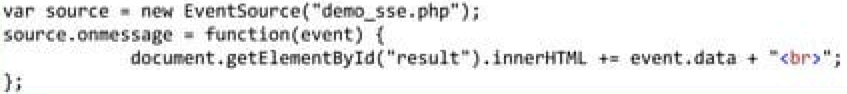
- The manifest file is modified
- The user clears the browser's cache
- The application cache is programmatically updated

**Note:** Be careful what you cache. Once a file is cached, the browser will continue to show the cached version, even if you change the file on the server. Also, browsers may have different size limits for cached data.

# SSE

- Stands for **Server-Sent Events**
- When a web page automatically gets updates from a server eg. Twitter, Facebook updates, news feeds.
- SSEs are supported in all major browsers, except Internet Explorer.

The **EventSource object** is used to receive server-sent event notifications:



```
var source = new EventSource("demo_sse.php");
source.onmessage = function(event) {
            document.getElementById("result").innerHTML += event.data + "<br>";
};
```

***What's going on:***

- We create a new EventSource object, specify the URL of the page sending updates (demo_sse.php)
- Each time an update is received, the onmessage event occurs
- When an onmessage event occurs, put the received data into the element with id="result"

# SSE Ctd.

- For the code to work, you need a server capable of sending data updates (php, asp).
- Set the “Content-Type” to “text/event-stream” to start sending streams.

*Code in PHP:*

```
<?php
header('Content-Type: text/event-stream');
header('Cache-Control: no-cache');
$time = date('r');
echo "data: The server time is: {$time}\n\n";
flush();
?>
```

*Code in ASP:*

```
<%
Response.ContentType = "text/event-stream"
Response.Expires = -1
Response.Write("data: The server time is: " & now())
Response.Flush()
%>
```

*What's going on:*

- Set the Content-Type header to “text/event-stream”
- Specify that the page should not cache
- Output the data to send
- Flush the output data back to the web page

# Useful Resources

- W3 Schools
- W3C
- Udemy
- HTML5 Rocks
- HTML5 Rocks Slides
- Lynda.com
- HTML Cheat Sheet
- Evolution of HTML