HTML5: Going Beyond the Static Website

Instructor: Donnie Seigler

Day One

I. HTML5 Semantic Elements

A. Semantic (from Ancient Greek) Elements are Elements with Meaning

1. **Clearly Describes It's Meaning to Both the Browser & Developer**
2. **Non-Semantic Elements: & - Tells Nothing About It's Content**
3. **Semantic Elements: <form>, <table>, & <img> - Clearly Defines It's Content**

B. Why Use Semantic Elements?

1. **With HTML4 Developers Used Custom Attribute Names**
2. **No Way for Search Engines to Identify Page Content Consistently**
3. **HTML5 Semantic Tags such as <header>, <nav> & <footer> Solve the Problem**

C. HTML5 Semantic Tags are Supported by All Major Browsers

1. **<article>**
    - ***a.*** **Specifies Independent, Self-contained Content**
    - ***b.*** **Should be Able to Stand Alone & Possible to Read Independently from Rest of Site**
    - ***c.*** **Use for Forum Posts, Blog Posts & News Articles**
2. **<aside>**
    - ***a.*** **Defines Content as an Aside from surrounding Content**
    - ***b.*** **Should be Related to the Surrounding Content**
    - ***c.*** **Great for Sidebar Content or Related Pull-outs in an Article**

3. **<details>**
    a. Specifies Additional Details Users Can View or Hide on Demand
    b. Create an Interactive Widget Users Can Open and Close
    c. The Open Attribute Specifies that Details Should be Visible to Users
    d. The **<summary>** tag Specifies a Visible Heading for the Details
    e. The **<summary>** Heading Can be Clicked to View/Hide Details
    f. Not Currently Supported by IE, Edge or Firefox
4. **<figcaption>**
    a. Defines a Caption for a **<figure>** Element
    b. Can be the First or Last Child of the **<figure>** Element
5. **<figure>**
    a. Specifies Self-contained Content Like Illustrations, Diagrams, Images, etc.
    b. The Content is Related to the Main Flow but It's Position is Independent of the Main Flow
    c. If Removed it Should Not Affect the Flow of the Document
6. **<footer>**
    a. Defines a Footer for a Document or Section
    b. Should Contain Information About It's Containing Element
    c. Can Have Multiple **<footer>** Elements in a Document
    d. Typically Includes Author Information, Copyright, Contact Information, Sitemap, Back to Top Links or Related Documents
    e. Contact Information Should go in an **<address>** Element
7. **<header>**
    a. Represents a Container for Introductory Content or a Set of Navigational Links
    b. Can Have Multiple **<header>** Elements in a Document
    c. A **<header>** Tag Cannot be Placed Within a **<footer>**, **<address>** or Another **<header>** Element

d. Typically Contains Heading Elements (<h1> - <h6>), Logos or Icons & Authorship Information

8. <main>
    a. Specifies the Main Content of a Document
    b. Content Inside the <main> Element Should be Unique to the Document
    c. Should Not Contain Any Content Repeated Across Documents Such as Sidebars, Menus, Copyright Information, Logos or Search Forms
    d. A Document Can Only Have One <main> Element
    e. The <main> Element Must NOT be a Descendent of an <article>, <aside>, <footer>, <header> or <nav> Element
    f. Not Supported by IE, but is Supported by Microsoft Edge
9. <mark>
    a. Defines Marked Text
    b. Use the <mark> Tag to Highlight Parts of Your Text
10. <nav>
    a. Defines a Set of Navigation Links or Menus
    b. Intended Only for Major Blocks of Navigation Links
    c. Can Have Multiple <nav> Elements in a Document
    d. Browsers, Such as Screen Readers for Disabled Users, Can Determine Whether to Omit the Initial Rendering of this Content
11. <section>
    a. Defines Sections in a Document
    b. Typically Used for Chapters, Headers, Footers or Any Other Sections of the Document
12. <summary>
    a. Defines a Visible Heading for the <details> Element
    b. The Heading Can be Clicked to View/Hide the Details
    c. Should be the First Child Element of the <details> Element

d. Not Supported by IE, Edge or Firefox

13. <time>

a. Defines a Human-Readable Date/Time

b. Can be Used to Encode Dates & Times in a Machine-Readable Way so User Agents Can Add Birthday Reminders or Scheduled Events to the User's Calendar & Search Engines Can Produce Smarter Search Results

c. The datetime Attribute Represent a Machine-Readable Date/Time of the <time> Element

d. Does Not Render Anything Special in Any Browser

D. Nesting Semantic Elements

1. In the HTML5 Standard, the <article> Element Defines a Complete, Self-contained Block of Related Elements
2. The <section> Element is Defined as a Block of Related Elements
3. Can We Use the Definitions to Decide How to Nest Elements? No, We Cannot!
4. On the Internet, You Will Find HTML Pages with <section> Elements Containing <article> Elements & <article> Elements Containing <sections> Elements
5. You Will Also Find Pages with <section> Elements Containing <section> Elements & <article> Elements Containing <article> Elements
6. Newspaper: The Sports Articles in the Sports Section, Have a Technical Section in Each Article

II. HTML5 Form Elements

A. Not All Browsers Support All the New Form Elements

B. If Not Supported They Will Behave as Regular Text Fields

C. <datalist>

1. specifies a List of Pre-defined Options for an <input> Element

2. Used as an "Autocomplete" Feature on <input> Elements
3. Users Will See a Drop-Down List of Pre-defined Options as They Input Data
4. Use the <input> Element's List Attribute to Bind it Together with a <datalist> Element
5. If the User Must Enter One of Your Choices, Use the <select> Element
6. If the User Can Enter Whatever, Use the <datalist> Element
7. Not Supported in Safari

D. <keygen>

1. The Purpose of the <keygen> Element is to Provide a Secure Way to Authenticate Users
2. Specifies a Key-Pair Generator Field in a Form
3. When the Form is Submitted, Two Keys are Generated, One Private and One Public
4. The Private Key is Stored Locally, and the Public Key is Sent to the Server
5. The Public Key Could be Used to Generate a Client Certificate to Authenticate the User in the Future
6. Can be Used to Improve Security in Combination with a Passphrase or Smartcard or as a Convenient Replacement for Entering a Password
7. Client Certificates are Bound to the Browser, so if on Another Computer You Will Not Have Access to Them
8. The <keygen> Element Has Six Attributes
    a. autofocus - Specifies that a <keygen> Element Should Automatically get Focus When the Page Loads
    b. challenge - Specifies that the Value of the <keygen> Element Should be Challenged When Submitted
    c. disabled - Specifies that a <keygen> Element Should be Disabled

d. form – (form_id) - Specifies One or More Forms the <keygen> Element Belongs to

e. eytype – (rsa, dsa, ec) - Specifies the Security Algorithm of the Key. RSA is Default

f. name - Defines a Name for the <keygen> Element

E. <output>

1. Represents the Result of a Calculation (like one performed by a script)
2. Displayed as an Inline Element by Default
3. The <output> Element Has Three Attributes
    a. for – (element_id) - Specifies the Relationship Between the Result of the Calculation & the Elements Used in the Calculation
    b. form – (form_id) - Specifies One or More Forms the <output> Element Belongs to
    c. name – (name) - Specifies a Name for the <output> Element
4. Not Supported in IE or Edge

III. HTML5 Form Input Types

A. Input types, not Supported by your web browser, will behave as regular text (input) types

B. Input Type: number

1. Used for Input Fields that Should Contain a Numeric Value
2. You Can Set Restrictions on the Numbers Using the min/max Attributes
3. Depending on Browser Support the Restrictions Can Apply to the Input Field

C. Input Type: date

1. used for Input Fields that Should Contain a Date
2. Depending on Browser Support a Date Picker Can Show up in the Input Field

3. Not Supported by IE or Firefox

D. Input Type: color

1. Used for Input Fields that Should Contain a Color
2. Depending on Browser Support, a Color Picker Can Show up in the Input Field
3. Not Supported by IE or Safari

E. Input Type: range

1. Used for Input Fields that Should Contain a Value Within a Range
2. Depending on Browser Support, the Input Field can be Displayed as a Slider Control
3. You Can Use the Following Attributes to Specify Restrictions: min, max, step, value

F. Input Type: month

1. Allows the User to Select a Month and Year
2. Depending on Browser Support, a Date Picker Can Show up in the Input Field
3. Not Supported by IE or Firefox

G. Input Type: week

1. Allows the User to Select a Week and Year
2. Depending on Browser Support, a Date Picker Can Show up in the Input Field
3. Not Supported by IE or Firefox

H. Input Type: time

1. Allows the User to Select a Time (no time zone)
2. Depending on Browser Support, a Time Picker Can Show up in the Input Field
3. Not Supported by IE or Firefox

I. **Input Type: datetime**

1. Allows the User to Select a Date and Time (with time zone)
2. Depending on Browser Support, a Date Picker Can Show up in the Input Field
3. Not Currently Supported by Any Browser

J. **Input Type: datetime-local**

1. Allows the User to Select a Date and Time (no time zone)
2. Depending on Browser Support, a Date Picker Can Show up in the Input Field
3. Not Supported by IE or Firefox

K. **Input Type: email**

1. Used for Input Fields that Should Contain an E-mail Address
2. Depending on Browser Support, the E-mail Address Can be Automatically Validated when Submitted
3. Some Smartphones Recognize the Email Type, and Adds ".com" to the Keyboard to Match Email Input
4. Not Supported by Safari

L. **Input Type: search**

1. Used for Search Fields (a search field behaves like a regular text field)
2. Nothing Special Occurs in the Browser
3. Not Supported by IE, Firefox or Opera

M. **Input Type: tel**

1. Used for Input Fields that Should Contain a Telephone Number
2. Currently Only Supported by Safari

N. **Input Type: url**

1. Used for Input Fields that Should Contain a URL Address

2. Depending on Browser Support, the URL Field Can be Automatically Validated when Submitted
3. Some Smartphones Recognize the URL Type, and Adds ".com" to the Keyboard to Match URL Input
4. Not Supported by Safari

IV. HTML5 Form Attributes

A. <form> / <input> autocomplete Attribute

1. Specifies Whether a <form> or <input> Should Have autocomplete Values of “on” or “off”
2. If "on" the Browser Automatically Completes Values Based Prior User Entries
3. It is Possible to Have "on" for the <form>, and "off" for Specific <input> Fields, or Vice Versa
4. Works with <input> Types: text, search, url, tel, email, password, datepickers, range & color
5. In Some Browsers You May Need to Activate the autocomplete Function for This to Work
6. Not Supported by Opera

B. <form> novalidate Attribute

1. The novalidate Attribute is a Boolean Attribute
2. When Present it Specifies that the <input> Should Not be Validated When Submitted
3. Not Supported by Safari

C. <input> autofocus Attribute

1. The autofocus Attribute is a Boolean Attribute
2. Specifies that an <input> Element Should Automatically get Focus When the Page Loads

3. If Used Multiple Times Only the First <input> Element Will get Focus

D. <input> form Attribute

1. Specifies One or More Forms an <input> Element Belongs To
2. To Refer to More than One Form Use a Space-separated List of Form IDs
3. An <input> Located Outside the HTML Form (but still a part of the form)
4. Not Supported in IE

E. <input> formaction Attribute

1. Specifies the URL of a File That Will Process the Input Control When the Form is Submitted
2. Overrides the action Attribute of the <form> Element
3. Only Used with Types submit & image
4. Example: an HTML Form with Two Submit Buttons with Different Actions

F. <input> formenctype Attribute

1. Specifies How the Form Data Should be Encoded When Submitting it to the Server (only for forms with method="post")
2. Overrides the enctype Attribute of the <form> Element
3. Only Used with Types submit & image

G. <input> formmethod Attribute

1. Defines the HTTP Method for Sending Form Data to the action URL
2. Overrides the method Attribute of the <form> Element
3. Only Used with Types submit & image

H. <input> formnovalidate Attribute

1. The novalidate Attribute is a Boolean Attribute
2. Specifies that the <input> Element Should Not be Validated When Submitted

3. Overrides the novalidate Attribute of the <form> Element
4. Only Used with Type submit
5. Not Supported by Safari

I. <input> formtarget Attribute

1. Specifies a Name or a Keyword That Indicates Where to Display the Response That is Received After Submitting the Form
2. Overrides the target Attribute of the <form> Element
3. Only Used with Types submit & image

J. <input> height and width Attributes

1. Specifies the height and width of an <input> Element
2. Only Used with Type image
3. Always Specify the Size of Images (if the browser does not know the size the page will flicker while images load)

K. <input> list Attribute

1. The list Attribute Refers to a <datalist> Element That Contains Pre-defined Options for an <input> Element
2. Not Supported by Safari

L. <input> min and max Attributes

1. Specifies the Minimum & Maximum Values for an <input> Element
2. Works with the Following Input Types: number, range, date, datetime, datetime-local, month, time and week
3. Not Supported by Firefox

M. <input> multiple Attribute

1. The multiple Attribute is a Boolean Attribute
2. Specifies That the User is Allowed to Enter More Than One Value in the <input> Element
3. Works with the Input Types email & file

N. **<input> pattern Attribute**

1. Specifies a Regular Expression That the <input> Element's Value is Checked Against
2. Works with the Following Input Types: text, search, url, tel, email & password
3. Use the Global title Attribute to Describe the Pattern to Help the User
4. Not Supported by Safari

O. **<input> placeholder Attribute**

1. Specifies a Hint That Describes the Expected Value of an Input Field (a sample value or a short description of the format)
2. The Hint is Displayed in the Input Field Before the User Enters a Value
3. Works with the Following Input Types: text, search, url, tel, email & password

P. **<input> required Attribute**

1. The required Attribute is a Boolean Attribute
2. Specifies That an Input Field Must be Filled Out Before Submitting the Form
3. Works with the Following Input Types: text, search, url, tel, email, password, date pickers, number, checkbox, radio & file
4. Not Supported by Safari

Q. **<input> step Attribute**

1. Specifies the Legal Number Intervals for an <input> Element
2. Example: If step="3", Legal Numbers Could be -3, 0, 3, 6, etc.
3. Can be Used Together with the max & min Attributes to Create a Range of Legal Values
4. Works with the Following Input Types: number, range, date, datetime, datetime-local, month, time & week

5. Not Supported by Firefox

Day Two

I. HTML5 Plugins

A. HTML Helpers or Plugins

1. A Plugin Extends the Functionality of the HTML Browser
2. Helper Applications are Computer Programs that Extend the Standard Functionality of a Web Browser
3. Helper Applications are also Known as Plugins, such as Java Applets
4. Plugins Can be Added to Web Pages with the <objec> Tag or the <embed> Tag
5. Plugins Can be Used for Many Purposes: Display Maps, Scan for Viruses, Verify Your Bank ID, etc.

B. The <objec> Element

1. The <objec> Element Defines an Embedded Object Within an HTML Document
2. Used to Embed Plugins Like Java Aapplets, PDF Readers, Flash Players in Web Pages
3. Can be Used to Embed HTML Snippets and Images as Well
4. Uses the data Attribute to Define the Path Name to the Resource
5. Has a Closing Tag and Can Include Alternative Text that will Display if the Tag is Not Supported by the Browser

C. The <embed> Element

1. The <embed> Element Also Defines an Embedded Object Within an HTML Document
2. Uses the src Attribute to Define the Path Name to the Resource

3. Browsers Have Supported the <embed> Element for Years, However, it was Not an HTML Standard Before HTML5
4. Note that the <embed> Element Does Not Have a Closing Tag nor Can it Contain Alternative Text

II. HTML5 Audio

A. Audio on the Web

1. Prior to HTML5 no Standard Existed for Playing Audio Files on a Web Page
2. Before HTML5 Audio Files Could Only be Played with a Plugin (like flash)
3. The HTML5 <audio> Element Specifies a Standard Way to Embed Audio in a Web Page

B. The <audio> Element

1. <audio> - Defines Sound Content
2. <source> - Defines Multiple Media Resources for Media Elements Such as <video> & <audio>

C. How It Works

1. Use the <audio> Element to Play Audio in a Web Page
2. The controls Attribute Adds Audio Controls Like Play, Pause, & Volume
3. Text Between the <audio> & </audio> Tags Will Display in Browsers that do Not Support the <audio> Element
4. Multiple <source> Elements Can Link to Different Audio Files, the Browser Will Use the First Recognized Format
5. Currently There are 3 Supported audio Formats for the <audio> Element
    a. MP3 (type="audio/mpeg") – All 5 Major Browsers
    b. WAV (type="audio/wav") – Chrome, Firefox, Opera & Safari
    c. Ogg (type="audio/ogg") – Chrome, Firefox & Opera

D. HTML Audio - Methods, Properties, & Events

1. HTML5 Defines DOM Methods, Properties & Events for the <audio> Element
2. This Allows You to Load, Play, & Pause Audios as Well as Setting Duration & Volume
3. There are Also DOM Events that Can Notify You When an Audio Begins to Play, is Paused, etc.

III. HTML5 Video

A. Playing Videos in HTML

1. Prior to HTML5 No Standard Existed for Showing Videos on a Web Page
2. Before HTML5 Videos Could Only be Played with a Plugin (like flash)
3. The HTML5 <video> Element Specifies a Standard Way to Embed a Video in a Web Page

B. HTML5 Video Tags

1. <video> - Defines a Video or Movie
2. <source> - Defines Multiple Media Resources for Media Elements Such as <video> & <audio>
3. <track> - Defines Text Tracks in Media Players

C. The <video> Element

1. To Show a Video in HTML Use the <video> Element
2. The controls Attribute Adds Video Controls Like Play, Pause & Volume
3. It is a Good Idea to Always Include width and height Attributes
4. If height & width are Not Set the Browser Can't Determine the Size of the Video Causing it to Flicker While the Video Loads
5. Text Between the <video> & </video> Tags Will Only Display in Browsers that do Not Support the <video> Element

6. Multiple <source> Elements Can Link to Different Video Files, the Browser Will Use the First Recognized Format
7. To Atart a Video Automatically Use the autoplay Attribute
8. Currently There are 3 Supported Video Formats for the <video> Element
    a. MP4 (type="video/mp4") – All 5 Major Browsers
    b. WebM (type="video/webm") – Chrome, Firefox and Safari
    c. Ogg (type="video/ogg") – Chrome, Firefox and Safari

D. HTML Video - Methods, Properties, & Events

1. HTML5 Defines DOM Methods, Properties & Events for the <video> Element
2. This Allows You to Load, Play & Pause Videos as Well as Set Duration & Volume
3. There are Also DOM Events that Can Notify You When a Video Begins to Play, is Paused, etc.

IV. HTML5 Geolocation Object

A. Locate the User's Position

1. The HTML Geolocation API is Used to Get the Geographical Position of a User
2. The Position is Not Available Unless the User Approves it Due to Security Concerns
3. Geolocation is Much More Accurate for Devices with GPS, Like iPhone
4. Geolocation is Very Useful for Location-specific Information
    a. Provide Up-to-date Local Information
    b. Showing Points-of-interest Near the User
    c. Turn-by-turn Navigation (GPS)
    d. Hiking/Climbing Apps to Record Distance & Changes in Altitude

B. Using HTML5 Geolocation

1. Access the Geolocation API with navigator.geolocation
2. Use getCurrentPosition() Method to Get the User's Position
3. The getCurrentPosition() Method Returns the Following Properties:
    a. coords.latitude - The Latitude as a Decimal Number
    b. coords.longitude - The Longitude as a Decimal Number
    c. coords.accuracy - The Accuracy of Position
    d. coords.altitude - The Altitude in Meters Above the Mean Sea Level
    e. coords.altitudeAccuracy - The Altitude Accuracy of Position
    f. coords.heading - The Heading as Degrees Clockwise from North
    g. coords.speed - The Speed in Meters per Second
    h. coords.timestamp - The Date/Time of the Response
4. The getCurrentPosition() Method Parameters
    a. Results Function
    b. Error Function
    c. Options
5. You Can Pass These Options in the Third Parameter:
    a. enableHighAccuracy – Provides a Hint that the Application Would Like the Best Possible Results. May Cause a Slower Response Time and in the Case of a Mobile Device Greater Power Consumption as it May Use GPS. Boolean with a Default Setting of False
    b. timeout – Indicates the Maximum Length of Time to Wait for a Response. In Milliseconds with a Default of 0 – Infinite
    c. maximumAge – Denotes the Maximum Age of a Cached Position that the Application Will be Willing to Accept. In Milliseconds with a Default Value of 0, Which Means that an Attempt Must be Made to Obtain a New Position Object Immediately

C. Handling Errors & Rejections

1. The Second Parameter of the getCurrentPosition() Method is Used to Handle Errors and Specifies a Function to Run if it Fails to Get the User's Location
2. Error Codes
    a. error.PERMISSION_DENIED - User Did Not Allow Geolocation
    b. error.POSITION_UNAVAILABLE - It is Not Possible to Get the Current Location
    c. error.PERMISSION_TIMEOUT - The Operation Timed Out
    d. error.UNKNOWN_ERROR – An Unknown Error

D. Displaying the Result in a Map

1. To Display the Result in a Map You Need Access to a Map Service that Can Use Latitude and Longitude Like Google Maps
2. Using Google You Can Display the Location Either on a Static Map or an Interactive Map

E. Geolocation Object - Other Interesting Methods

1. watchPosition() - Returns the Current Position of the User and Continues to Return Updated Position as the User Moves Like the GPS in a Car
2. clearWatch() - Stops the watchPosition() Method
3. You Need an Accurate GPS Device to Test this Like an iPhone

V. HTML5 Drag and Drop

A. What is Drag and Drop?

1. It is When You "Grab" an Object and "Drag" it to a Different Location and "Drop" it Using Your Mouse
2. Drag and Drop is Part of the HTML5 Standard
3. Any Element Can be Set as Draggable
4. There are Several Uses of Drag and Drop

a. Drag Links to the Browser Address Bar or Bookmarks

b. Drag Files onto the Desktop or Client File System

c. Drag Images to Complete a Puzzle

B. Make an Element Draggable

1. Set the draggable Attribute to True on the Element You Want to Drag
2. Ex. <img draggable="true">

C. What to Drag - ondragstart and setData()

1. Set the ondragstart Attribute to Call Your Drag Function to Specify What Should Happen When the Element is Dragged
2. Ex. <img draggable="true" ondragstart="drag(event)">
3. Specify the Data to be Dragged in JavaScript
   a. Use ev.dataTransfer.setData(dataType, value)
   b. dataType is Usually Set to "text" but Other Types Such as URL are Available
   c. value is Usually Set the the Draggable Element's ID Using ev.target.id but Can be a String Such as an URL

D. Where to Drop - ondragover

1. The ondragover Event Specifies Where the Dragged Data Can be Dropped
2.
3. By Default Data/Elements Cannot be Dropped in Other Elements
4. To Allow a Drop We Must Prevent the Default Handling of the Element
5. This is Done by Calling the event.preventDefault() Method for the ondragover Event

E. Do the Drop - ondrop

1. When the Dragged Data is Dropped a Drop Event Occurs
2. Use the ondrop Attribute to Call Your Drop Function to Append the Dragged Element to the Specified Drop Element

3. Ex.
4. In Your Drop Function:
    a. Call preventDefault() to Prevent the Browser Default Handling of the Data (default is open as link on drop)
    b. Get the Dragged Data with the dataTransfer.getData() Method
    c. This method Will Return Any Data that was Set to the Same Type in the setData() Method
    d. The Dragged Data is the ID of the Dragged Element
    e. Append the Dragged Element into the Drop Element with ev.target.appendChild(document.getElementById(ID))

VI. HTML5 Local Storage

A. What is HTML Local Storage?

1. With Local Storage Web Applications Can Store Data Locally Within the User's Browser
2. Before HTML5 Application Data had to be Stored in Cookies Included in Every Server Request
3. Local Storage is More Secure and Large Amounts of Data Can be Stored Locally Without Affecting Website Performance
4. Unlike Cookies the Storage Limit is Far Larger (at least 5MB) and Information is Never Transferred to the Server
5. Local Storage is Per Domain, All Pages from One Domain Can Store and Access the Same Data
6. Supported in All Browsers

B. HTML Local Storage Objects

1. HTML Local Storage Provides Two Objects for Storing Data on the Client

a. window.localStorage - Stores Data with No Expiration Date (data will not be deleted when the browser is closed and will be available the next day, week or year)

b. code.sessionStorage - Stores Data for One Session (data is lost when the tab is closed)

2. Before Using Local Storage Check Browser Support for localStorage & sessionStorage
3. Name/Value Pairs are Always Stored as Strings so Remember to Convert them to Another Format When Needed!

VII. HTML5 Server-Sent Events

A. Server-Sent Events - One Way Messaging

1. A Server-Sent Event is When a Web Page Automatically Gets Updates from a Server
2. This was Also Possible Before but the Web Page Would Have to Ask if Any Updates were Available
3. With Server-Sent Events the Updates Come Automatically
4. Examples:
    a. Facebook/Twitter Updates
    b. Stock Price Updates
    c. News Feeds
    d. Sport Results
5. Not Supported by Internet Explorer

B. Receive Server-Sent Event Notifications

1. The EventSource Object is Used to Receive Server-Sent Event Notifications
2. Create a New EventSource Object and Specify the URL of the Page Sending the Updates
    a. `var source = new EventSource("demo_sse.php");`

3. Each Time an Update is Received the onmessage Event Occurs
    a. source.onmessage = function(event)
4. When an onmessage Event Occurs Put the Received Data into the Element with id="result"
    a. document.getElementById("result").innerHTML += event.data + "
";

C. Check Server-Sent Events Support

1. Use an if...else Statement to Check for Server-Sent Events
2. Ex. if(typeof(EventSource) !== "undefined")
3. If Supported Add Code for the SSE
4. If Not Supported Add Code for Alternative Action

D. The EventSource Object Events

1. onopen - When a Connection to the Server is Opened
2. onmessage - When a Message is Received
3. onerror - When an Error Occurs

E. Server-Side Code

1. you Need a Server Capable of Sending Data Updates (like PHP or ASP)
2. Set the "Content-Type" Header to "text/event-stream"
3. Specify that the Page Should Not Cache
4. Output the Data to Send (always start with "data: ")
5. Flush the Output Data Back to the Web Page