**HTML Reference**

<!DOCTYPE html>

* start document with this tag
* designates type of document
* tells browser to use latest version of html

<html></html>

* use after doctype tag
* contain all document information
* **lang** attribute to designate language (first 2 letters) and dialect (last two letters)

<head></head>

* use to collect metadata for document
* open and close tag before <body> tag

<title></title>

* used in <head> tag
* designates title of document

<body></boy>

* contains all visible content in document

<h1></h1>......<h2></h2>......<h3></h3>...

* heading tags
* h1 most important, h2 next most, h3 and so on until h6

<p></p>

* paragraph tag
* **title** attribute provides title that appears when hovered over

<ul></ul>

* unordered list tag (bullet points)

<ol></ol>

* ordered list tag (numbered)

<li></li>

* list item contained (nested) within list tags

<a></a>

* anchor tag used for hyperlinking
* **href** attribute for source address
* **target** attribute “\_blank” opens in new tab; “\_self” opens in current tab; “\_parent” opens link in parent frame; “\_top” opens link in full body of window

<img></img>

* image tag
* **src** attribute for source address
* **width** attribute for width of image
* **height** attribute for height of image
* **alt** attribute displays text when image can’t be seen

<br>

* line break

When content contains double quotes, start with single quotes (vice versa)

Important Attributes

* alt – alternative text when image does not display
* disabled – specifies that input element should be disabled
* href – specifies URL
* id – specifies unique id for element
* src – URL for image
* style – specifies CSS style for an element
* title – extra info about an element (displayed as tool tip)

<hr>

* creates horizontal line in text

Metadata – data about HTML document contained in the <head></head> tags

<pre></pre>

* defines preformatted text (fixed width font) and preserves space and line breaks

CSS Format - <tagname style=“property:value;”>

* property – CSS property
* value – CSS value

Important CSS Properties

* background-color (behind text)
* color (text color)
* font-family (text font)
* font-size (text size)
* text-align (text alignment)

<b></b>

* bold text

<strong></strong>

* important text (also shows bold)

<i></i>

* italics

<em></em>

* emphasize text (also shows italics)

<mark></mark>

* highlights text

<small></small>

* makes text small

<del></del>

* puts line through text

<ins></ins>

* underlines text

<sub></sub>

* turns text to subscript

<sup></sup>

* turns text to superscript

<q></q>

* used for short quotes, adds quotation marks

<blockquote></blockquote>

* defines a quote from another source
* held in black quote format

<abbr></abbr>

* defines abbreviations/acronyms
* **title** attribute used to define the abbreviation

<address></address>

* element defines contact information of document or an article
* displayed in italics, line break will occur before and after

<cite></cite>

* defines title of a work, appear italics

<bdo></bdo>

* bi-directional override, give new direction to text
* **dir** attribute allows new direction (ex. right to left is “rtl”)

<!--..........-->

* comment tag, does not affect page

Color can be specified by color name, RGB value, or HEX value

* RGB Value – rbg(0,0,0)
  + The values can vary between 0 and 255
* Hex Value #RRGGBB
  + The values can vary between 00 and FF (same as 0 to 255)

**CSS** – Cascading Style Sheets

* Can be added to HTML by:
  + inline – using style attributes in HTML elements
  + internal – using <style> within the <head> section
  + external – using an external CSS file
    - Use <link rel=“stylesheet” href=“file.css”
* inline overrides internal which overrides external

CSS Format:

h1 {

color: red;

background-color:white;

}

CSS Features:

* color – text color
* font-family – text font
* font-size – defines text size
* border – border around HTML element
* padding – space between text and border
* margin – space outside border

Add an *id* to a tag to add an attribute(s) to a single element

ex. <p id= “p01”>...</p>

#p01 {

color:red;

}

Add a *class* to a tag to add an attribute(s) to multiple elements

ex. <li class= “tab”>...</li>

li.tab {

color:red;

}

Hyperlinks color attributes added in CSS

* a:link { } color of regular hyperlink
* a:visited { } color of hyperlink after clicked
* a:hover { } color of hyperlink when hovered over
* a:active { } color of hyperlink when active

Images can be hyperlinked by:

<a href= “...”>

<img>

</a>

Bookmarks

<h2 id= “BM1”>...</h2>

<a href= “#BM1”>...</a> 🡨 when in same doc

<a href= “file.html#BM1”>...</a> 🡨 when in different doc

Use float attribute to have image float to right or left of text.

<img src= “...” alt= “...” float=”right”>

<map><area></map>

* creates an image map

<table></table>

* used to develop tables
* <tr> defines a table row
* <th> defines a table header
* <td> defines a table data input

CSS table attributes:

* border: 1px solid black;
* border-collapse: collapse; to get normal looking table
* text-align: left; align text within cell
* border-spacing:1px; spacing of inner borders
* colspan = “2” – merge two cells in columns
* rowspan = “2” – merge two cells in row

<caption></caption>

* add title to table

Vertical Lists:

* Unordered list: <ul style= “list-style-type:disc;”></ul>
* Ordered list: <ol type= “1”>
* Description lists <dl>...</dl>
  + <dt></dt>
  + <dd></dd>

Horizontal List (tab menu):

<head>

<style>

ul { }

li { }

li a { }

li a:hover { }

</style>

</head>

Block Element – starts on new line, takes up full width of screen available

* ex. <div> <h1> <p> <form>

Inline Elements – does not start on new line and only takes up as much width as necessary

* ex. <span> <a> <img>

<iframe></iframe>

* used to display different webpages and files
* <iframe src= “...” height= “...” width= “...” style= “border:none;”>
* can be used as a target frame (to display different images upon clicking a button

ex. <iframe src= “...” name= “pic1”></iframe>

<a href= “...” target= “pic1”>...</a>

<script><script>

* used to define a client-side script (JavaScript)
* either contains scripting statements or directs to an external script file
* selects an HTML element by using document.getElementBy(id)

ex. <script>

function Function( ) {

document.getElementById(“...”)

}

</script>

<button type= “button” onclick= “Function(True)”></button>

<button type= “button” onclick= “Function(False)”></button>

<noscript></noscript>

* alternative content for users that have disabled scripts in their browser

File paths:

* <img src= “pic.jpg”> - picture is located in same folder as current page
* <img src= “image/pic.jpg”> - picture is located in the image folder located in the current folder
* <img src= “/image/pic.jpg”> - picture is located in the images folder ar the root of the current web
* <img src= “../pic.jpg”> - picture is located in the folder one level up from the current folder
* Absolute – uses full URL
* Relative – points to a file relative to current page
* Using relative file path is best practice

<head></head>

* contains metadata that typically defines document title, character set, styles, links, scripts and other information
  + <title>, <style>, <meta>, <link>, <script>, <base>
* placed before <body>

<title></title>

* defines title in browser tab and provides title for page

<link></link>

* used to link to external style sheets

<meta>

* used to specify which character set is used, page description, keywords, author,
* metadata is used by search engines/browsers
* <meta charset= “UTF-8”> - defines character set used
* <meta name= “description” content= “Free Web tutorials”> - define description of your web page
* <meta name= “keywords” content= “HTML, CSS, XML, JavaScript”> - define keywords for search engines
* <meta name= “author” content= “Kristian Schif”> - author of document
* <meta http-equiv= “refresh” content= “30”> - refresh document every 30 seconds

Viewport

* viewport – users visible area of a web page. It varies with the device, and will be smaller on a mobile phone than on a computer screen
* <meta name= “viewport” content= “width=device-width, initial-scale=1.0”>
* width=device-width sets the width of the page to follow the screen-width of the device
* initial-scale=1.0 sets the initial zoom level when the page is first loaded by the browser

<base>

* specifies base URL and base target for all relative URLs in a page
* <base href= “https://kristianschif.org/” target= “\_blank”>

Layouts

* <header> - defines a header for a document of a section
* <nav> - defines a container for navigation links
* <section> - defines a section in a document
* <article> - defines an independent self-contained article
* <aside> - defines a content aside from the content (like a sidebar)
* <footer> - defines a footer for a document or a section
* <details> - defines additional details
* <summary> - defines a heading for the <details> element

Layout Techniques:

* HTML tables – used to display tabular data; do not use for layouts
* CSS float properties
* CSS framework
* CSS flexbox – use this to display layouts; accommodates for different display devices; does not work in Internet Explorer 10 or earlier

Responsive Web Design

* makes it look good on all devices
* create them yourself
* can use W3.css demo

<kbd></kbd>

* represents user input
* displayed in browser as monospace font

<samp></samp>

* represents output form a program or computing system
* displayed in monospace font

<code></code>

* defines a fragment of computer code
* displayed in monospace font

<pre></pre>

* preserves line breaks and extra spaces

<var></var>

* defines a variable
* could be a variable in a mathematical expression or a variable in programming context

Entities – names are used to display different characters

* ex. < symbol would get confused with a tag, so **&lt;** or **&#60;** is used for < symbol

Result Description Entity Name Entity Number

non-breaking space &nbsp; &#160;

< less than &lt; &#60;

> greater than &gt; &#62;

& ampersand &amp; &#38;

" double quotation &quot; &#34;

' single quotation &apos; &#39;

¢ cent &cent; &#162;

£ pound &pound; &#163;

¥ yen &yen; &#165;

€ euro &euro; &#8364;

© copyright &copy; &#169;

® registered trademark &reg; &#174;

Combining Diacritical Marks

* diacritical mark is a “glyph” added to a letter

Mark Character Construct Result

̀ a a&#768; à

́ a a&#769; á

̂ a a&#770; â

̃ a a&#771; ã

̀ O O&#768; Ò

́ O O&#769; Ó

̂ O O&#770; Ô

̃ O O&#771; Õ

HTML Symbol Entities

* use to add mathematical, technical, and currency symbols (ace, diamonds, up arrow, etc.)

HTML Encoding (Character Sets)

* UTF-8 (Unicode) covers almost all of the characters and symbols in the world
* Use UTF-8

URL

* another word for a web address
* can be composed of words (kristianschif.com) or an Internet Protocol (IP) address (192.68.20.50)
* URL – Uniform Resource Locator
* URL’s follow these syntax rules
  + scheme – defines the type of Internet service (http or https)
  + prefix – defines domain prefix (default for http is www)
  + domain – defines internet domain (kristianschif)
  + port – defines the port number at the host (default for http is 80)
  + path – defines a path at the server (if omitted: the root directory of the site)
  + filename – defines the name of a document or resource

URL Schemes

* http – hypertext transfer protocol – common web pages, not encrypted
* https – hypertext transfer protocol secure – secure web pages, encrypted
* ftp – file transfer protocol – downloading or uploading files
* file – a file on your computer
* fills spaces in a URL with either + or %20

XHTML – eXtensible HyperText Markup Language

* almost identical to HTML except a little stricter
  + must follow all proper format

<form></form>

* defines a form that is used to collect user input
* contains form elements (text fields, checkboxes, radio buttons, spin buttons, etc.)

<input></input>

* used to input different features based on the type of attribute
* <input type= “text”> - one-line text input field
* <input type= “radio> - defines a radio button (for selecting one of many choices)
* <input type= “submit”> - defines a submit button (for submitting a form)

action attribute defines the action to be performed when form is submitted

* <form action= “/action\_page.php”>

Get vs Post

* <form action= “/action\_page.php” method= “get”> - use when sending short or non-sensitive info
* <form action= “/action\_page.php” method= “post”> - use when sending long or sensitive info

Name attribute to <input>

* if the name attribute is omitted, the data of that input field will not be sent
* ex. <input type=”text” name= “lastname” value= “Schif”>

Value attribute fills input with default value

<fieldset></fieldset>

* used to group related data in a form

<legend></legend>

* defines a caption for the <fieldset> element

Sending email from form

* <form action="mailto:someone@example.com" method="post" enctype="text/plain">

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| accept-charset | Specifies the charset used in the submitted form (default: the page charset). |
| action | Specifies an address (url) where to submit the form (default: the submitting page). |
| autocomplete | Specifies if the browser should autocomplete the form (default: on). |
| enctype | Specifies the encoding of the submitted data (default: is url-encoded). |
| method | Specifies the HTTP method used when submitting the form (default: GET). |
| name | Specifies a name used to identify the form (for DOM usage: document.forms.name). |
| novalidate | Specifies that the browser should not validate the form. |
| target | Specifies the target of the address in the action attribute (default: \_self). |

<select></select>

* Defines a drop-down list

<select name= “cars”>

<option value= “volve”> Volvo </option>

</select>

<option></option>

* defines an option that can be selected
* selected attribute can be used to pre-select an option

<textarea name= “...” rows= “10” cols= “30”></textarea>

* used as a large textbox

<button type= “button” onclick= “alert(‘Hello!’)”></button>

* clickable button

<datalist></datalist>

* specifies a list of pre-defined options for an <input> element
* user will see drop-down list of pre-defined options as they input data

<keygen>

* a secure way to authenticate users
* private key is stored locally, public key is sent to the server

<output></output>

* represents the result of a calculation

<input type= “password”>

* defines a password field
* shown as circles or asterisks

<input type= “submit”>

* defines a button for submitting data to a form-handler

<input type= “submit”>

* reset all form values to their default values

<input type= “radio”>

* lets users select limited number of choices

<input type= “checkbox”>

* allows user to select zero to more options

Different input types:

* color – input color from pallete
* date – calendar feature
* datetime-local – calendar and time
* email – validates email when submitted
* month – calendar for month and year
* number – defines a numeric input field
* range – slider control bar
* search – behaves similar to a regular text field
* tel – contain a telephone number
* time – allows user to select a time
* url – validates URL upon submission
* week – week and year input

Input attributes:

* value – initial value of field
* readonly – specifies that input field is read only (cannot be changed)
* disabled – specifies that input field is diabled
* size – specifies size (In characters)
* maxlength – specifies maximum allowed length for input
* autocomplete – specifies whether a form should have autocomplete on or not (automatically complete based on previous values)
* novalidate – specifies that the form data should not be validated when submitted
* autofocus – specifies that the input field should automatically get focus when the page loads (highlights field)
* form – specifies one or more forms an <input> element belongs to
* formaction – specifies the URL of a file that will process the input control when the form is submitted; overrides the action attribute of the <form> element
* formenctype – specifies how the form data should be encoded when submitted (only for forms with method= “post”)
* formmethod – defines the HTTP method for sending form-data to the action URL
* formnovalidate – overrides the novalidate attribute of the <form> element
* formtarget – specifies a name or keyboard that indicates where to display the response that is received after submitting the form
* height – height of field
* width – width of field
* min and max – specifies min and max values for an input element
* multiple – specifies that the user is allowed to enter more than one value
* pattern – specifies a regular expression that the <input> element’s value is checked against
* placeholder – specifies a hint that describes the expected value of an input field
* required – specifies that an input field must be filled out before submitting the form
* step – specifies the legal number intervals for an input element

DOCTYPE for HTML5 is <!DOCTYPE html>

Character encoding (charset) <meta charset= “UTF-8”>

New HTML5 elements:

* semantic elements - <header>, <footer>, <article>, <section>
* attributes of form elements – number, date, time, calendar, range
* graphic elements - <svg>, <canvas>
* multimedia elements - <audio>, <video>

API – application programming interface:

* HTML Geolocation
* HTML Drag and Drop
* HTML Local Storage
* HTML Application Cache
* HTML Web Workers
* HTML SSE

Elements can be added to HTML

ex. <script> document.createElement(“Test”)</script>

<style>

Test {

display: block;

background-color: #ffffff;

}

Use the HTML5Shiv.js file when using the new HTML5 elements

New HTML5 Semantic/Structural Elements

|  |  |
| --- | --- |
| **Tag** | **Description** |
| <article> | Defines an article in a document |
| <aside> | Defines content aside from the page content |
| <bdi> | Isolates a part of text that might be formatted in a different direction from other text outside it |
| <details> | Defines additional details that the user can view or hide |
| <dialog> | Defines a dialog box or window |
| <figcaption> | Defines a caption for a <figure> element |
| <figure> | Defines self-contained content |
| <footer> | Defines a footer for a document or section |
| <header> | Defines a header for a document or section |
| <main> | Defines the main content of a document |
| <mark> | Defines marked/highlighted text |
| <menuitem> | Defines a command/menu item that the user can invoke from a popup menu |
| <meter> | Defines a scalar measurement within a known range (a gauge) |
| <nav> | Defines navigation links |
| <progress> | Represents the progress of a task |
| <rp> | Defines what to show in browsers that do not support ruby annotations |
| <rt> | Defines an explanation/pronunciation of characters (for East Asian typography) |
| <ruby> | Defines a ruby annotation (for East Asian typography) |
| <section> | Defines a section in a document |
| <summary> | Defines a visible heading for a <details> element |
| <time> | Defines a date/time |
| <wbr> | Defines a possible line-break |

New HTML5 Form Elements

|  |  |
| --- | --- |
| **Tag** | **Description** |
| <datalist> | Specifies a list of pre-defined options for input controls |
| <keygen> | Defines a key-pair generator field (for forms) |
| <output> | Defines the result of a calculation |

HTML5 Graphics

|  |  |
| --- | --- |
| <canvas> | Draw graphics, on the fly, via scripting (usually JavaScript) |
| <svg> | Draw scalable vector graphics |

New Media Elements

|  |  |
| --- | --- |
| **Tag** | **Description** |
| <audio> | Defines sound content |
| <embed> | Defines a container for an external (non-HTML) application |
| <source> | Defines multiple media resources for media elements (<video> and <audio>) |
| <track> | Defines text tracks for media elements (<video> and <audio>) |
| <video> | Defines video or movie |

Semantics is the study of the meanings of words and phrases in a language

* non-semantic - <div><span> tells you nothing about its content
* semantic - <form><table> clearly defines its content

New HTML5 semantic elements:

* <article>
* <aside>
* <details>
* <figcaption>
* <figure>
* <footer>
* <header>
* <main>
* <mark>
* <nav>
* <section>
* <summary>
* <time>

<section></section>

* defines a section in a document

<article></article>

* specifies independent, self-contained content
* should make sense on its own, and it should be possible to read it independently from the rest of the web site
  + blog post
  + newspaper article

<figure><img><figcaption></figcaption></figure>

* used to contain all image info

You can use .dtd file to convert between HTML4 and HTML5

Important things to remember:

* use lowercase at all times
* close empty html elements

ex. <meta charset= “UTF-8” />

* always add alt attribute to images
* Use least amount of spaces as possible

<canvas is used to draw graphics on the fly via JavaScript

* <canvas> element is only a container for graphics
* JavaScript must be used to draw the graphics

SVG – scalable vector graphics

* <svg> element is a container for SVG graphics

Canvas vs SVG

* Canvas is resolution dependent (like .jpg)
* SVG is resolution independent

Tutorial on how to add a google map [here](https://www.w3schools.com/html/html_googlemaps.asp)

Multimedia – files such as sound, videos, etc.

Video formats

|  |  |  |
| --- | --- | --- |
| **Format** | **File** | **Description** |
| MPEG | .mpg .mpeg | MPEG. Developed by the Moving Pictures Expert Group. The first popular video format on the web. Used to be supported by all browsers, but it is not supported in HTML5 (See MP4). |
| AVI | .avi | AVI (Audio Video Interleave). Developed by Microsoft. Commonly used in video cameras and TV hardware. Plays well on Windows computers, but not in web browsers. |
| WMV | .wmv | WMV (Windows Media Video). Developed by Microsoft. Commonly used in video cameras and TV hardware. Plays well on Windows computers, but not in web browsers. |
| QuickTime | .mov | QuickTime. Developed by Apple. Commonly used in video cameras and TV hardware. Plays well on Apple computers, but not in web browsers. (See MP4) |
| RealVideo | .rm .ram | RealVideo. Developed by Real Media to allow video streaming with low bandwidths. It is still used for online video and Internet TV, but does not play in web browsers. |
| Flash | .swf .flv | Flash. Developed by Macromedia. Often requires an extra component (plug-in) to play in web browsers. |
| Ogg | .ogg | Theora Ogg. Developed by the Xiph.Org Foundation. Supported by HTML5. |
| WebM | .webm | WebM. Developed by the web giants, Mozilla, Opera, Adobe, and Google. Supported by HTML5. |
| MPEG-4 or MP4 | .mp4 | MP4. Developed by the Moving Pictures Expert Group. Based on QuickTime. Commonly used in newer video cameras and TV hardware. Supported by all HTML5 browsers. Recommended by YouTube. |

Audio Formats

|  |  |  |
| --- | --- | --- |
| **Format** | **File** | **Description** |
| MIDI | .mid .midi | MIDI (Musical Instrument Digital Interface). Main format for all electronic music devices like synthesizers and PC sound cards. MIDI files do not contain sound, but digital notes that can be played by electronics. Plays well on all computers and music hardware, but not in web browsers. |
| RealAudio | .rm .ram | RealAudio. Developed by Real Media to allow streaming of audio with low bandwidths. Does not play in web browsers. |
| WMA | .wma | WMA (Windows Media Audio). Developed by Microsoft. Commonly used in music players. Plays well on Windows computers, but not in web browsers. |
| AAC | .aac | AAC (Advanced Audio Coding). Developed by Apple as the default format for iTunes. Plays well on Apple computers, but not in web browsers. |
| WAV | .wav | WAV. Developed by IBM and Microsoft. Plays well on Windows, Macintosh, and Linux operating systems. Supported by HTML5. |
| Ogg | .ogg | Ogg. Developed by the Xiph.Org Foundation. Supported by HTML5. |
| MP3 | .mp3 | MP3 files are actually the sound part of MPEG files. MP3 is the most popular format for music players. Combines good compression (small files) with high quality. Supported by all browsers. |
| MP4 | .mp4 | MP4 is a video format, but can also be used for audio. MP4 video is the upcoming video format on the internet. This leads to automatic support for MP4 audio by all browsers. |

<video></video>

* can now be used to embed videos in HTML5

ex. <video width="320" height="240" controls>

<source src="movie.mp4" type="video/mp4">

<source src="movie.ogg" type="video/ogg">

Your browser does not support the video tag.

</video>

* autoplay attribute will automatically start the video
* stick to .mp4 if possible

<audio></audio>

* specifies a standard way to embed audio in a web page
* stick to .mp3 if possible

Plug-ins

* helper applications that extend the standard functionality of a web browser
* <object> defines an embedded object within the document
* <embed> does a similar thing that <object> does

Youtube Videos

* videos contain ids (such as XGSy3\_Czz8k)
* define an <iframe> element with src attribute point to URL
* src=”https:...../XGSy3\_Czz8k**?autoplay=1”** makes the video autoplay
* **&loop=1”** makes it loop forever
* **?controls=0”** does not display media controls

Geolocation API – used to locate a user’s position

getCurrentPosition() Method

|  |  |
| --- | --- |
| **Property** | **Returns** |
| coords.latitude | The latitude as a decimal number (always returned) |
| coords.longitude | The longitude as a decimal number (always returned) |
| coords.accuracy | The accuracy of position (always returned) |
| coords.altitude | The altitude in meters above the mean sea level (returned if available) |
| coords.altitudeAccuracy | The altitude accuracy of position (returned if available) |
| coords.heading | The heading as degrees clockwise from North (returned if available) |
| coords.speed | The speed in meters per second (returned if available) |
| timestamp | The date/time of the response (returned if available) |

Make an Element Draggable

* <img draggable= “true”>

ondragstart attribute <img ondragstart= “...”> calls a function drag(event) that specifies what data to be dragged

dataTransfer.setData() sets the data type and the value of the dragged data

ex. function drag(ev) {

ev.dataTransfer.setData(“text”, ev.target.id);

}

ondragover event specifies where the dragged data can be dropped; be sure to call the event.preventDefault()

function drop(ev) {

ev.preventDefault();

var data = ev.dataTransfer.getData(“text”);

ev.target.appendChild(document.getElementById(data));

}

Local storage – storing data locally within the user’s browser

* before HTML5, data had to be stored in cookies, included in every server request; local storage is more secure and large amounts of data can be stored locally, without affecting website performance

window.localStorage – stores data with no expiration date

window.sessionStorage – stores data for one session (data is lost when the browser tab is closed

To check for browser support of localStorage and sessionStorage use the following function:

if (typeof(Storage) !== “undefined”) {

// Code for localStorage/sessionStorage.

} else {

// Sorry! No web storage support..

}

// Store

localStorage.setItem("lastname", "Smith");

// Retrieve

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

Web Worker – a JavaScript running in the background, without affecting the performance of the page

To check for browser support of web workers, use the following function:

if (typeof(Worker) !== "undefined") {

// Yes! Web worker support!

// Some code.....

} else {

// Sorry! No Web Worker support..

}

SSE – server-sent events; a web page automatically gets updates from a server (Facebook, Twitter)

EventSource object is used to receive server-sent event notifications

To check for browser support of server-sent events, use the following function:

if(typeof(EventSource) !== "undefined") {

// Yes! Server-sent events support!

// Some code.....

} else {

// Sorry! No server-sent events support..

}