Cascading style sheets means

I'm not robot!

CSS Selectors

Selector	Role
p{ }	Tag selector, **all p tags**
#para{ }	Id para (***unique***)
.para1{ }	Class para1 (***multiple***)
p.para{ }	P tag with class para
P .para{}	P with child having class para
div p{}	Div with child p tag
*{}	All tags{ **Universal Selector**}
h1, h3, h5{}	Only h1, h3 and h5 (***grouping***)
.para a{ }	A with parent para class
body{}	Parent of all tags

Why it is called cascading style sheets. What does it mean that the style sheets are cascading. How do cascading style sheets work. What is meant by cascading style sheets.

View Discussion Improve Article Save Article Like Article Cascading Style Sheet(CSS) is used to set the style in web pages that contain HTML elements. It sets the background color, font-size, font-family, color, ... etc property of elements on a web page. There are three types of CSS which are given below: Inline CSSInternal or Embedded CSSExternal CSSInline CSS: Inline CSS contains the CSS property in the body section attached with element is known as inline CSS. This kind of style is specified within an HTML tag using the style attribute. Example: Inline CSS GeeksForGeeks Output: Internal or Embedded CSS: This can be used when a single HTML document must be styled uniquely. The CSS rule set should be within the HTML file in the head section i.e the CSS is embedded within the HTML file. Example: Internal CSS .main { text-align:center; } .GFG { color:#009900; font-size:50px; font-weight:bold; } .geeks { font-style:bold; font-size:20px; } GeeksForGeeks A computer science portal for geeks Output: External CSS: External CSS contains separate CSS file which contains only style property with the help of tag attributes (For example class, id, heading, ... etc). CSS property written in a separate file with .css extension and should be linked to the HTML document using link tag. This means that for each element, style can be set only once and that will be applied across web pages.Example: The file given below contains CSS property. This file save with .css extension. For Ex: geeks.css body { background-color:powderblue; } .main { text-align:center; } .GFG { color:#009900; font-size:50px; font-weight:bold; } #geeks { font-style:bold; font-size:20px; }Below is the HTML file that is making use of the created external style sheet link tag is used to link the external style sheet with the html webpage.href attribute is used to specify the location of the external style sheet file. GeeksForGeeks A computer science portal for geeks Output: Properties of CSS: Inline CSS has the highest priority, then comes Internal/Embedded followed by External CSS which has the least priority. Multiple style sheets can be defined on one page. If for an HTML tag, styles are defined in multiple style sheets then the below order will be followed. As Inline has the highest priority, any styles that are defined in the internal and external style sheets are overridden by Inline styles.Internal or Embedded stands second in the priority list and overrides the styles in the external style sheet.External style sheets have the least priority. If there are no styles defined either in inline or internal style sheet then external style sheet rules are applied for the HTML tags.Supported Browser:Google ChromeInternet ExplorerFirefoxOperaSafariCSS is the foundation of webpages, is used for webpage development by styling websites and web apps.You can learn CSS from the ground up by following this CSS Tutorial and CSS Examples. The definition of CSS on this page is an original TechTerms.com definition. If you would like to reference this page or cite this definition, you can use the green citation links above. The goal of TechTerms.com is to explain computer terminology in a way that is easy to understand. We strive for simplicity and accuracy with every definition we publish. If you have feedback about the CSS definition or would like to suggest a new technical term, please contact us. Want to learn more tech terms? Subscribe to the daily or weekly newsletter and get featured terms and quizzes delivered to your inbox. Find out how AT Internet will empower you to skyrocket your acquisition, conversion and retention rates. Our advanced and powerful solution is trusted by 1000s of our customers, including, the BBC, Le Monde and Total. Drive your web analytics into the fast lane! Educative Answers TeamTransforming the webpage using CSS1 of 5Copyright ©2022 Educative, Inc. All rights reserved More examples Nearly all browsers nowadays support CSS and many other applications do, too. To write CSS, you don't need more than a text editor, but there are many tools available that make it even easier. Of course, all software has bugs, even after several updates. And some programs are further ahead implementing the latest CSS modules than others. Various sites describe bugs and work-arounds. More » For beginners, Starting with HTML + CSS teaches how to create a style sheet. For a quick introduction to CSS, try chapter 2 of Lie & Bos or Dave Raggett's intro to CSS. Or see examples of styling XML and CSS tips & tricks. Another page also has some books, mailing lists and similar fora, and links to other directories. The history of CSS is described in chapter 20 of the book Cascading Style Sheets, designing for the Web, by Håkon Wium Lie and Bert Bos (2nd ed., 1999, Addison Wesley, ISBN 0-201-59625-3) More » Site navigation Style sheet language "Pseudo-element" redirects here. For pseudoelement symbols in chemistry, see Skeletal formula § Pseudoelement symbols. For other uses, see CSS (disambiguation). For the use of CSS on Wikipedia, see Help:Cascading Style Sheets. Cascading Style Sheets (CSS)The official logo of the latest version, CSS 3Filename extension .cssInternet media type text/cssUniform Type Identifier (UTI)public.cssDeveloped byWorld Wide Web Consortium (W3C)Initial release17 December 1996; 25 years ago (1996-12-17)Latest releaseCSS 2.1 : Level 2 Revision 112 April 2016; 6 years ago (2016-04-12) Type of formatStyle sheet languageContainer forStyle rules for HTML elements (tags)Contained byHTML DocumentsOpen format? YesWebsitewww.w3.org/TR/CSS/#css Cascading Style Sheets Style sheet CSS Zen Garden Concepts animations box model image replacement flexbox grid Philosophies Tableless Responsive "Holy grail" Tools Sass Less Stylus CSSTidy Comparisons CSS support Stylesheet languages Cascading Style Sheetsvte HTML Dynamic HTML HTML5 audio canvas video XHTML Basic Mobile Profile C-HTML HTML element span and div HTML attribute HTML frame HTML editor Character encodings named characters Unicode Language code Document Object Model Browser Object Model Style sheets CSS Font family Web colors JavaScript WebGL WebCL W3C Validator WHATWG Quirks mode Web storage Rendering engine Comparisons Document markup languages HTML support XHTML 1.1 vte Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML or XML (including XML dialects such as SVG, MathML or XHTML).[1] CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.[2] CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts.[3] This separation can improve content accessibility; provide more flexibility and control in the specification of presentation characteristics; enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, which reduces complexity and repetition in the structural content; and enable the .css file to be cached to improve the page load speed between the pages that share the file and its formatting. Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device.[4] The name cascading comes from the specified priority scheme to determine which style rule applies if more than one rule matches a particular element. This cascading priority scheme is predictable. The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) text/css is registered for use with CSS by RFC 2318 (March 1998). The W3C operates a free CSS validation service for CSS documents.[5] In addition to HTML, other markup languages support the use of CSS including XHTML, plain XML, SVG, and XUL. Syntax CSS has a simple syntax and uses a number of English keywords to specify the names of various style properties. A style sheet consists of a list of rules. Each rule or rule-set consists of one or more selectors, and a declaration block. Selector In CSS, selectors declare which part of the markup a style applies to by matching tags and attributes in the markup itself. Selectors may apply to the following: all elements of a specific type, e.g. the second-level headers h2 elements specified by attribute, in particular: id: an identifier unique within the document, identified with a hash prefix e.g. #id class: an identifier that can annotate multiple elements in a document, identified with a period prefix e.g. .classname elements depending on how they are placed relative to others in the document tree. Classes and IDs are case-sensitive, start with letters, and can include alphanumeric characters, hyphens, and underscores. A class may apply to any number of instances of any elements. An ID may only be applied to a single element. Pseudo-classes are used in CSS selectors to permit formatting based on information that is not contained in the document tree. One example of a widely used pseudo-class is :hover, which identifies content only when the user "points to" the visible element, usually by holding the mouse cursor over it. It is appended to a selector as in a:hover or #elementid:hover. A pseudo-class classifies document elements, such as :link or :visited, whereas a pseudo-element makes a selection that may consist of partial elements, such as ::first-line or ::first-letter.[6] Note the double-colon notation for pseudo-elements versus single-colon notation for pseudo-class. Selectors may be combined in many ways to achieve great specificity and flexibility.[7] Multiple selectors may be joined in a spaced list to specify elements by location, element type, id, class, or any combination thereof. The order of the selectors is important. For example, div .myClass {color: red;} applies to all elements of class myClass that are inside div elements, whereas .myClass div {color: red;} applies to all div elements that are inside elements of class myClass. This is not to be confused with concatenated identifiers such as div.myClass {color: red;} which applies to div elements of class myClass. The following table provides a summary of selector syntax indicating usage and the version of CSS that introduced it.[8] Pattern Matches First definedin CSS level E an element of type E 1 E:link an E element is the source anchor of a hyperlink of which the target is not yet visited (:link) or already visited (:visited) 1 E:active an E element during certain user actions 1 E::first-line the first formatted line of an E element 1 E::first-letter the first formatted letter of an E element 1 .c all elements with class="c" 1 #myid the element with id="myid" 1 E.warning an E element whose class is "warning" (the document language specifies how class is determined) 1 E#myid an E element with ID equal to "myid" 1 .c#myid the element with class="c" and ID equal to "myid" 1 E F an F element descendant of an E element 1 * any element 2 E[foo] an E element with a "foo" attribute 2 E[foo="bar"] an E element whose "foo" attribute value is exactly equal to "bar" 2 E[foo~="bar"] an E element whose "foo" attribute value is a list of whitespace-separated values, one of which is exactly equal to "bar" 2 E[foo|="en"] an E element whose "foo" attribute has a hyphen-separated list of values beginning (from the left) with "en" 2 E:first-child an E element, first child of its parent 2 E:lang(fr) an element of type E in language "fr" (the document language specifies how language is determined) 2 E::before generated content before an E element's content 2 E::after generated content after an E element's content 2 E > F an F element child of an E element 2 E + F an F element immediately preceded by an E element 2 E[foo^="bar"] an E element whose "foo" attribute value begins exactly with the string "bar" 3 E[foo$="bar"] an E element whose "foo" attribute value ends exactly with the string "bar" 3 E[foo*="bar"] an E element whose "foo" attribute value contains the substring "bar" 3 E:root an E element, root of the document 3 E:nth-child(n) an E element, the n-th child of its parent 3 E:nth-last-child(n) an E element, the n-th child of its parent, counting from the last one 3 E:nth-of-type(n) an E element, the n-th sibling of its type 3 E:nth-last-of-type(n) an E element, the n-th sibling of its type, counting from the last one 3 E:last-child an E element, last child of its parent 3 E:first-of-type an E element, first sibling of its type 3 E:last-of-type an E element, last sibling of its type 3 E:only-child an E element, only child of its parent 3 E:only-of-type an E element, only sibling of its type 3 E:empty an E element that has no children (including text nodes) 3 E:target an E element being the target of the referring URI 3 E:enabled a user interface element E that is enabled 3 E:disabled a user interface element E that is disabled 3 E:checked a user interface element E that is checked (for instance a radio-button or checkbox) 3 E:not(s) an E element that does not match simple selector s 3 E ~ F an F element preceded by an E element 3 Declaration block A declaration block consists of a list of declarations in braces. Each declaration itself consists of a property, a colon (:), and a value. If there are multiple declarations in a block, a semi-colon (;) must be inserted to separate each declaration. An optional semi-colon after the last (or single) declaration may be used.[9] Properties are specified in the CSS standard. Each property has a set of possible values. Some properties can affect any type of element, and others apply only to particular groups of elements.[10][11] Values may be keywords, such as "center" or "inherit", or numerical values, such as 200px (200 pixels), 50vw (50 percent of the viewport width) or 80% (80 percent of the parent element's width). Color values can be specified with keywords (e.g. "red"), hexadecimal values (e.g. #FF0000, also abbreviated as #F00), RGB values on a 0 to 255 scale (e.g. rgb(255, 0, 0)), RGBA values that specify both color and alpha transparency (e.g. rgba(255, 0, 0, 0.8)), or HSL or HSLA values (e.g. hsl(000, 100%, 50%), hsla(000, 100%, 50%, 80%)).[12] Length units Non-zero numeric values representing linear measures must include a length unit, which is either an alphabetic code or abbreviation, as in 200px or 50vw; or a percentage sign, as in 80%. Some units – cm (centimetre); in (inch); mm (millimetre); pc (pica); and pt (point) – are absolute, which means that the rendered dimension does not depend upon the structure of the page; others – em (em); ex (ex) and px (pixel)[clarification needed] – are relative, which means that factors such as the font size of a parent element can affect the rendered measurement. These eight units were a feature of CSS 1[13] and retained in all subsequent revisions. The proposed CSS Values and Units Module Level 3 will, if adopted as a W3C Recommendation, provide seven further length units: ch; Q; rem; vh; vmax; vmin; and vw.[14] Use Before CSS, nearly all presentational attributes of HTML documents were contained within the HTML markup. All font colors, background styles, element alignments, borders and sizes had to be explicitly described, often repeatedly, within the HTML. CSS lets authors move much of that information to another file, the style sheet, resulting in considerably simpler HTML. For example, headings (h1 elements), sub-headings (h2), sub-sub-headings (h3), etc., are defined structurally using HTML. In print and on the screen, choice of font, size, color and emphasis for these elements is presentational. Before CSS, document authors who wanted to assign such typographic characteristics to, say, all h2 headings had to repeat HTML presentational markup for each occurrence of that heading type. This made documents more complex, larger, and more error-prone and difficult to maintain. CSS allows the separation of presentation from structure. CSS can define color, font, text alignment, size, borders, spacing, layout and many other typographic characteristics, and can do so independently for on-screen and printed views. CSS also defines non-visual styles, such as reading speed and emphasis for aural text readers. The W3C has now deprecated the use of all presentational HTML markup.[15] For example, under pre-CSS HTML, a heading element defined with red text would be written as: Chapter 1. Using CSS, the same element can be coded using style properties instead of HTML presentational attributes: Chapter 1. The advantages of this may not be immediately clear but the power of CSS becomes more apparent when the style properties are placed in an internal style element or, even better, an external CSS file. For example, suppose the document contains the style element: h1 { color: red; } All h1 elements in the document will then automatically become red without requiring any explicit code. If the author later wanted to make h1 elements blue instead, this could be done by changing the style element to: h1 { color: blue; } rather than by laboriously going through the document and changing the color for each individual h1 element. The styles can also be placed in an external CSS file, as described below, and loaded using syntax similar to: This further decouples the styling from the HTML document and makes it possible to restyle multiple documents by simply editing a shared external CSS file. Sources CSS information can be provided from various sources. These sources can be the web browser, the user, and the author. The information from the author can be further classified into inline, media type, importance, selector specificity, rule order, inheritance, and property definition. CSS style information can be in a separate document, or it can be embedded into an HTML document. Multiple style sheets can be imported. Different styles can be applied depending on the output device being used; for example, the screen version can be quite different from the printed version, so that authors can tailor the presentation appropriately for each medium. The style sheet with the highest priority controls the content display. Declarations not set in the highest priority source are passed on to a source of lower priority, such as the user agent style. The process is called cascading. One of the goals of CSS is to allow users greater control over presentation. Someone who finds red italic headings difficult to read may apply a different style sheet. Depending on the browser and the web site, a user may choose from various style sheets provided by the designers, or may remove all added styles and view the site using the browser's default styling, or may override just the red italic heading style without altering other attributes. CSS priority scheme (highest to lowest) Priority CSS source type Description 1 Importance The "!important" annotation overwrites the previous priority types 2 Inline A style applied to an HTML element via HTML "style" attribute 3 Media Type A property definition applies to all media types, unless a media specific CSS is defined 4 User defined Most browsers have the accessibility feature: a user defined CSS 5 Selector specificity A specific contextual selector (#heading p) overwrites generic definition 6 Rule order Last rule declaration has a higher priority 7 Parent inheritance If a property is not specified, it is inherited from a parent element 8 CSS property definition in HTML document CSS rule or CSS inline style overwrites a default browser value 9 Browser default The lowest priority: browser default value is determined by W3C initial value specifications Specificity Specificity refers to the relative weights of various rules.[16] It determines which styles apply to an element when more than one rule could apply. Based on specification, a simple selector (e.g. H1) has a specificity of 1, class selectors have a specificity of 1,0, and ID selectors a specificity of 1,0,0. Because the specificity values do not carry over as in the decimal system, commas are used to separate the "digits"[17] (a CSS rule having 11 elements and 11 classes would have a specificity of 11,11, not 121). Thus the following rules selectors result in the indicated specificity: Selectors Specificity h1 {color: white;} 0, 0, 0, 1 p em {color: green;} 0, 0, 0, 2 .grape {color: red;} 0, 0, 1, 0 p.bright {color: blue;} 0, 0, 1, 1 p.bright em.dark {color: yellow;} 0, 0, 2, 2 #id218 {color: brown;} 0, 1, 0, 0 style=" " 1, 0, 0, 0 Examples Consider this HTML fragment: #xyz { color: blue; } To demonstrate specificity In the above example, the declaration in the style attribute overrides the one in the element because it has a higher specificity, and thus, the paragraph appears green: To demonstrate specificity Inheritance Inheritance is a key feature in CSS; it relies on the ancestor-descendant relationship to operate. Inheritance is the mechanism by which properties are applied not only to a specified element, but also to its descendants.[16] Inheritance relies on the document tree, which is the hierarchy of XHTML elements in a page based on nesting. Descendant elements may inherit CSS property values from any ancestor element enclosing them. In general, descendant elements inherit text-related properties, but their box-related properties are not inherited. Properties that can be inherited are color, font, letter-spacing, line-height, list-style, text-align, text-indent, text-transform, visibility, white-space and word-spacing. Properties that cannot be inherited are background, border, display, float and clear, height, and width, margin, min- and max-height and -width, outline, overflow, padding, position, text-decoration, vertical-align and z-index. Inheritance can be used to avoid declaring certain properties over and over again in a style sheet, allowing for shorter CSS. Inheritance in CSS is not the same as inheritance in class-based programming languages, where it is possible to define class B as "like class A, but with modifications".[18] With CSS, it is possible to style an element with "class A, but with modifications". However, it is not possible to define a CSS class B like that, which could then be used to style multiple elements without having to repeat the modifications. Example Given the following style sheet: p { color: pink; } Suppose there is a p element with an emphasizing element () inside: This is to illustrate inheritance If no color is assigned to the em element, the emphasized word "illustrate" inherits the color of the parent element, p. The style sheet p has the color pink, hence, the em element is likewise pink: This is to illustrate inheritance Whitespace Whitespace between properties and selectors is ignored. This code snippet: body{overflow:hidden;background:#000000;background-image:url(images/bg.gif);background-repeat:no-repeat;background-position:left top;} is functionally equivalent to this one: body { overflow: hidden; background-color: #000000; background-image: url(images/bg.gif); background-repeat: no-repeat; background-position: left top; } One common way to format CSS for readability is to indent each property and give it its own line. In addition to formatting CSS for readability, shorthand properties can be used to write out the code faster, which also gets processed more quickly when being rendered:[19] body { overflow: hidden; background: #000 url(images/bg.gif) no-repeat left top; } Sometimes, multiple property values are indented onto their own line:@font-face { font-family: 'Comic Sans' font-size: 20px src: url('first.example.com'), url('second.example.com'), url('third.example.com'), url('fourth.example.com'), } Positioning CSS 2.1 defines three positioning schemes: Normal flow Inline items are laid out in the same way as the letters in words in text, one after the other across the available space until there is no more room, then starting a new line below. Block items stack vertically, like paragraphs and like the items in a bulleted list. Normal flow also includes relative positioning of block or inline items, and run-in boxes. Floats A floated item is taken out of the normal flow and shifted to the left or right as far as possible in the space available. Other content then flows alongside the floated item. Absolute positioning An absolutely positioned item has no place in, and no effect on, the normal flow of other items. It occupies its assigned position in its container independently of other items.[20] Position property There are five possible values of the position property. If an item is positioned in any way other than static, then the further properties top, bottom, left, and right are used to specify offsets and positions.The element having position static is not affected by the top, bottom , left or right properties. Static The default value places the item in the normal flow Relative The item is placed in the normal flow, and then shifted or offset from that position. Subsequent flow items are laid out as if the item had not been moved. Absolute Specifies absolute positioning. The element is positioned in relation to its nearest non-static ancestor. Fixed The item is absolutely positioned in a fixed position on the screen even as the rest of the document is scrolled[20] Float and clear The float property may have one of three values. Absolutely positioned or fixed items cannot be floated. Other elements normally flow around floated items, unless they are prevented from doing so by their clear property. left The item floats to the left of the line that it would have appeared in; other items may flow around its right side. right The item floats to the right of the line that it would have appeared in; other items may flow around its left side. clear Forces the element to appear underneath ('clear') floated elements to the left (clear:left), right (clear:right) or both sides (clear:both).[20][21] History Håkon Wium Lie, chief technical officer of the Opera Software company and co-creator of the CSS web standards CSS was first proposed by Håkon Wium Lie on 10 October 1994.[22] At the time, Lie was working with Tim Berners-Lee at CERN.[23] Several other style sheet languages for the web were proposed around the same time, and discussions on public mailing lists and inside World Wide Web Consortium resulted in the first W3C CSS Recommendation (CSS1)[24] being released in 1996. In particular, a proposal by Bert Bos was influential; he became co-author of CSS1, and is regarded as co-creator of CSS.[25] Style sheets have existed in one form or another since the beginnings of Standard Generalized Markup Language (SGML) in the 1980s, and CSS was developed to provide style sheets for the web.[26] One requirement for a web style sheet language was for style sheets to come from different sources on the web. Therefore, existing style sheet languages like DSSSL and FOSI were not suitable. CSS, on the other hand, let a document's style be influenced by multiple style sheets by way of "cascading" styles.[26] As HTML grew, it came to encompass a wider variety of stylistic capabilities to meet the demands of web developers. This evolution gave the designer more control over site appearance, at the cost of more complex HTML. Variations in web browser implementations, such as ViolaWWW and WorldWideWeb,[27] made consistent site appearance difficult, and users had less control over how web content was displayed. The browser/editor developed by Tim Berners-Lee had style sheets that were hard-coded into the program. The style sheets could therefore not be linked to documents on the web.[23] Robert Cailliau, also of CERN, wanted to separate the structure from the presentation so that different style sheets could describe different presentation for printing, screen-based presentations, and editors.[27] Improving web presentation capabilities was a topic of interest to many in the web community and nine different style sheet languages were proposed on the www-style mailing list.[26] Of these nine proposals, two were especially influential on what became CSS: Cascading HTML Style Sheets[22] and Stream-based Style Sheet Proposal (SSP).[25][28] Two browsers served as testbeds for the initial proposals; Lie worked with Yves Lafon to implement CSS in Dave Raggett's Arena browser.[29][30][31] Bert Bos implemented his own SSP proposal in the Argo browser.[25] Thereafter, Lie and Bos worked together to develop the CSS standard (the 'H' was removed from the name because these style sheets could also be applied to other markup languages besides HTML).[23] Lie's proposal was presented at the "Mosaic and the Web" conference (later called WWW2) in Chicago, Illinois in 1994, and again with Bert Bos in 1995.[23] Around this time the W3C was already being established, and took an interest in the development of CSS. It organized a workshop toward that end chaired by Steven Pemberton. This resulted in W3C adding work on CSS to the deliverables of the HTML editorial review board (ERB). Lie and Bos were the primary technical staff on this aspect of the project, with additional members, including Thomas Reardon of Microsoft, participating as well. In August 1996, Netscape Communication Corporation presented an alternative style sheet language called JavaScript Style Sheets (JSSS).[23] The spec was never finished, and is deprecated.[32] By the end of 1996, CSS was ready to become official, and the CSS level 1 Recommendation was published in December. Development of HTML, CSS, and the DOM had all been taking place in one group, the HTML Editorial Review Board (ERB). Early in 1997, the ERB was split into three working groups: HTML Working group, chaired by Dan Connolly of W3C; DOM Working group, chaired by Lauren Wood of SoftQuad; and CSS Working group, chaired by Chris Lilley of W3C. The CSS Working Group began tackling issues that had not been addressed with CSS level 1, resulting in the creation of CSS level 2 on November 4, 1997. It was published as a W3C Recommendation on May 12, 1998. CSS level 3, which was started in 1998, is still under development as of 2014. In 2005, the CSS Working Groups decided to enforce the requirements for standards more strictly. This meant that already published standards like CSS 2.1, CSS 3 Selectors, and CSS 3 Text were pulled back from Candidate Recommendation to Working Draft level. Difficulty with adoption This section needs to be updated. Please help update this article to reflect recent events or newly available information. (January 2019) The CSS 1 specification was completed in 1996. Microsoft's Internet Explorer 3[23] was released in that year, featuring some limited support for CSS. IE 4 and Netscape 4.x added more support, but it was typically incomplete and had many bugs that prevented CSS from being usefully adopted. It was more than three years before any web browser achieved near-full implementation of the specification. Internet Explorer 5.0 for the Macintosh, shipped in March 2000, was the first browser to have full (better than 99 percent) CSS 1 support,[33] surpassing Opera, which had been the leader since its introduction of CSS support fifteen months earlier. Other browsers followed soon afterward, and many of them additionally implemented parts of CSS 2.[citation needed] However, even when later "version 5" web browsers began to offer a fairly full implementation of CSS, they were still incorrect in certain areas and were fraught with inconsistencies, bugs and other quirks. Microsoft Internet Explorer 5.x for Windows, as opposed to the very different IE for Macintosh, had a flawed implementation of the CSS box model, as compared with the CSS standards. Such inconsistencies and variation in feature support made it difficult for designers to achieve a consistent appearance across browsers and platforms without the use of workarounds termed CSS hacks and filters. The IE Windows box model bugs were so serious that, when Internet Explorer 6 was released, Microsoft introduced a backwards-compatible mode of CSS interpretation ("quirks mode") alongside an alternative, corrected "standards mode". Other non-Microsoft browsers also provided mode-switch capabilities. It therefore became necessary for authors of HTML files to ensure they contained special distinctive "standards-compliant CSS intended" marker to show that the authors intended CSS to be interpreted correctly, in compliance with standards, as opposed to being intended for the now long-obsolete IE5/Windows browser. Without this marker, web browsers that have the "quirks mode"-switching capability will size objects in web pages as IE 5 on Windows would, rather than following CSS standards.[citation needed] Problems with patchy adoption of CSS, along with errata in the original specification, led the W3C to revise the CSS 2 standard into CSS 2.1, which moved nearer to a working snapshot of current CSS support in HTML browsers. Some CSS 2 properties that no browser successfully implemented were dropped, and in a few cases, defined behaviors were changed to bring the standard into line with the predominant existing implementations. CSS 2.1 became a Candidate Recommendation on February 25, 2004, but CSS 2.1 was pulled back to Working Draft status on June 13, 2005,[34] and only returned to Candidate Recommendation status on July 19, 2007.[35] In addition to these problems, the .css extension was used by a software product used to convert PowerPoint files into Compact Slide Show files,[36] so some web servers served all .css[37] as MIME type application/x-pointplus[38] rather than text/css. Vendor prefixes Individual browser vendors occasionally introduced new parameters ahead of standardization and universalization. To prevent interfering with future implementations, vendors prepended unique names to the parameters, such as -moz- for Mozilla Firefox, -webkit- named after the browsing engine of Apple Safari, -o- for Opera Browser and -ms- for Microsoft Internet Explorer and early versions of Microsoft Edge that use EdgeHTML. Occasionally, the parameters with vendor prefix such as -moz-radial-gradient and -webkit-linear-gradient have slightly different syntax as compared to their non-vendor-prefix counterparts.[39] Prefixed properties are rendered obsolete by the time of standardization. Programs are available to automatically add prefixes for older browsers, and to point out standardized versions of prefixed parameters. Since prefixes are limited to a small subset of browsers, removing the prefix allows other browsers to see the functionality. An exception is certain obsolete -webkit- prefixed properties, which are so common and persistent on the web that other families of browsers have decided to support them for compatibility.[40] Variations CSS Snapshot 2021 CSS has various levels and profiles. Each level of CSS builds upon the last, typically adding new features and typically denoted[citation needed] as CSS 1, CSS 2, CSS 3, and CSS 4. Profiles are typically a subset of one or more levels of CSS built for a particular device or user interface. Currently there are profiles for mobile devices, printers, and television sets. Profiles should not be confused with media types, which were added in CSS 2. CSS 1 The first CSS specification to become an official W3C Recommendation is CSS level 1, published on 17 December 1996. Håkon Wium Lie and Bert Bos are credited as the original developers.[41][42] Among its capabilities are support for Font properties such as typeface and emphasis Color of text, backgrounds, and other elements Text attributes such as spacing between words, letters, and lines of text Alignment of text, images, tables and other elements Margin, border, padding, and positioning for most elements Unique identification and generic classification of groups of attributes The W3C no longer maintains the CSS 1 Recommendation.[43] CSS 2 CSS level 2 specification was developed by the W3C and published as a recommendation in May 1998. A superset of CSS 1, CSS 2 includes a number of new capabilities like absolute, relative, and fixed positioning of elements and z-index, the concept of media types, support for aural style sheets (which were later replaced by the CSS 3 speech modules)[44] and bidirectional text, and new font properties such as shadows. The W3C no longer maintains the CSS 2 recommendation.[45] CSS 2.1 CSS level 2 revision 1, often referred to as "CSS 2.1", fixes errors in CSS 2, removes poorly supported or not fully interoperable features and adds already implemented browser extensions to the specification. To comply with the W3C Process for standardizing

technical specifications, CSS 2.1 went back and forth between Working Draft status and Candidate Recommendation status for many years. CSS 2.1 first became a Candidate Recommendation on 25 February 2004, but it was reverted to a Working Draft on 13 June 2005 for further review. It returned to Candidate Recommendation on 19 July 2007 and then updated twice in 2009. However, because changes and clarifications were made, it again went back to Last Call Working Draft on 7 December 2010. CSS 2.1 went to Proposed Recommendation on 12 April 2011.[46] After being reviewed by the W3C Advisory Committee, it was finally published as a W3C Recommendation on 7 June 2011.[47] CSS 2.1 was planned as the first and final revision of level 2—but low priority work on CSS 2.2 began in 2015. CSS 3 "CSS3" redirects here. For other uses, see CSS3 (disambiguation). Unlike CSS 2, which is a large single specification defining various features, CSS 3 is divided into several separate documents called "modules". Each module adds new capabilities or extends features defined in CSS 2, preserving backward compatibility. Work on CSS level 3 started around the time of publication of the original CSS 2 recommendation. The earliest CSS 3 drafts were published in June 1999.[48] Due to the modularization, different modules have different stability and statuses.[49] Some modules have Candidate Recommendation (CR) status and are considered moderately stable. At CR stage, implementations are advised to drop vendor prefixes.[50] This article needs to be updated. Please help update this article to reflect recent events or newly available information. (January 2021) Summary of main module-specifications[51] Module Specification title Status Date css3-background CSS Backgrounds and Borders Module Level 3 Candidate Rec. Dec 2020 css3-box CSS Box Model Module Level 3 Candidate Rec. Dec 2020 css-cascade-3 CSS Cascading and Inheritance Level 3 Recommendation Feb 2021 css3-color CSS Color Module Level 3 Recommendation Jun 2018 css3-content CSS Generated Content Module Level 3 Working Draft 2 Aug 2019 css-fonts-3 CSS Fonts Module Level 3 Recommendation Sep 2018 css3-gcpm CSS Generated Content for Paged Media Module Working Draft May 2014 css3-layout CSS Template Layout Module Note Mar 2015 css3-mediaqueries Media Queries Recommendation Jun 2012 mediaqueries-4 Media Queries Level 4 Candidate Rec. Jul 2020 css3-multicol Multi-column Layout Module Level 1 Working Draft Feb 2021 css3-page CSS Paged Media Module Level 3 Working Draft, and part migrated to css3-break Oct 2018 css3-break CSS Fragmentation Module Level 3 Candidate Rec. Dec 2018 selectors-3 Selectors Level 3 Recommendation Nov 2018 selectors-4 Selectors Level 4 Working Draft Nov 2018 css3-ui CSS Basic User Interface Module Level 3 (CSS3 UI) Recommendation Jun 2018 CSS 4 "CSS4" redirects here. For other uses, see CSS4 (disambiguation). Jen Simmons discussing the state of CSS in 2019, as several CSS 4 modules were being advanced There is no single, integrated CSS4 specification,[52] because the specification has been split into many separate modules which level independently. Modules that build on things from CSS Level 2 started at Level 3. Some of them have already reached Level 4 or are already approaching Level 5. Other modules that define entirely new functionality, such as Flexbox,[53] have been designated as Level 1 and some of them are approaching Level 2. The CSS Working Group sometimes publishes "Snapshots", a collection of whole modules and parts of other drafts that are considered stable enough to be implemented by browser developers. So far, five such "best current practices" documents have been published as Notes, in 2007,[54] 2010,[55] 2015,[56] 2017,[57] and 2018.[58] Since these specification snapshots are primarily intended for developers, there has been growing demand for as similar versioned reference document targeted at authors, which would present the state of interoperable implementations as meanwhile documented by sites like Can I Use...[59] and the MDN Web Docs.[60] A W3C Community Group has been established in early 2020 in order to discuss and define such a resource.[61] The actual kind of versioning is also up to debate, which means that the document once produced might not be called "CSS4". Browser support Further information: Comparison of web browser engines (CSS support) Each web browser uses a layout engine to render web pages, and support for CSS functionality is not consistent between them. Because browsers do not parse CSS perfectly, multiple coding techniques have been developed to target specific browsers with workarounds (commonly known as CSS hacks or CSS filters). Adoption of new functionality in CSS can be hindered by lack of support in major browsers. For example, Internet Explorer was slow to add support for many CSS 3 features, which slowed adoption of those features and damaged the browser's reputation among developers. Additionally, a proprietary syntax for the non-vendor-prefixed filter property was used in some versions.[62] In order to ensure a consistent experience for their users, web developers often test their sites across multiple operating systems, browsers, and browser versions, increasing development time and complexity. Tools such as BrowserStack have been built to reduce the complexity of maintaining these environments. In addition to these testing tools, many sites maintain lists of browser support for specific CSS properties, including CanIUse and the MDN Web Docs. Additionally, the CSS 3 defines feature queries, which provide an @supports directive that will allow developers to target browsers with support for certain functionality directly within their CSS.[63] CSS that is not supported by older browsers can also sometimes be patched in using JavaScript polyfills, which are pieces of JavaScript code designed to make browsers behave consistently. These workarounds—and the need to support fallback functionality—can add complexity to development projects, and consequently, companies frequently define a list of browser versions that they will and will not support. As websites adopt newer code standards that are incompatible with older browsers, these browsers can be cut off from accessing many of the resources on the web (sometimes intentionally).[64] Many of the most popular sites on the internet are not just visually degraded on older browsers due to poor CSS support, but do not work at all, in large part due to the evolution of JavaScript and other web technologies. Limitations Some noted limitations of the current capabilities of CSS include: Selectors are unable to ascend CSS currently offers no way to select a parent or ancestor of an element that satisfies certain criteria.[65] CSS Selectors Level 4, which is still in Working Draft status, proposes such a selector,[66] but only as part of the complete "snapshot" selector profile, not the fast "live" profile used in dynamic CSS styling.[67] A more advanced selector scheme (such as XPath) would enable more sophisticated style sheets. The major reasons for the CSS Working Group previously rejecting proposals for parent selectors are related to browser performance and incremental rendering issues.[68] Cannot explicitly declare new scope independently of position Scoping rules for properties such as z-index look for the closest parent element with a position:absolute or position:relative attribute. This odd coupling has undesired effects. For example, it is impossible to avoid declaring a new scope when one is forced to adjust an element's position, preventing one from using the desired scope of a parent element. Pseudo-class dynamic behavior not controllable CSS implements pseudo-classes that allow a degree of user feedback by conditional application of alternate styles. One CSS pseudo-class, ":hover", is dynamic (equivalent of JavaScript "onmouseover") and has potential for misuse (e.g., implementing cursor-proximity popups),[69] but CSS has no ability for a client to disable it (no "disable"-like property) or limit its effects (no "nochange"-like values for each property). Cannot name rules There is no way to name a CSS rule, which would allow (for example) client-side scripts to refer to the rule even if its selector changes. Cannot include styles from a rule into another rule CSS styles often must be duplicated in several rules to achieve a desired effect, causing additional maintenance and requiring more thorough testing. Some new CSS features were proposed to solve this, but were abandoned afterwards.[70][71] Instead, authors may gain this ability by using more sophisticated stylesheet languages which compile to CSS, such as Sass, Less, or Stylus. Cannot target specific text without altering markup Besides the :first-letter pseudo-element, one cannot target specific ranges of text without needing to utilize place-holder elements. Former issues Additionally, several more issues were present in prior versions of the CSS standard, but have been alleviated: Vertical control limitations Though horizontal placement of elements was always generally easy to control, vertical placement was frequently unintuitive, convoluted, or outright impossible. Simple tasks, such as centering an element vertically or placing a footer no higher than bottom of the viewport required either complicated and unintuitive style rules, or simple but widely unsupported rules.[65] The Flexible Box Module improved the situation considerably and vertical control is much more straightforward and supported in all of the modern browsers.[72] Older browsers still have those issues, but most of those (mainly Internet Explorer 9 and below) are no longer supported by their vendors.[73] Absence of expressions There was no standard ability to specify property values as simple expressions (such as margin-left: 10% – 3em + 4px;). This would be useful in a variety of cases, such as calculating the size of columns subject to a constraint on the sum of all columns. Internet Explorer versions 5 to 7 support a proprietary expression() statement,[74] with similar functionality. This proprietary expression() statement is no longer supported from Internet Explorer 8 onwards, except in compatibility modes. This decision was taken for "standards compliance, browser performance, and security reasons".[74] However, a candidate recommendation with a calc() value to address this limitation has been published by the CSS WG[75] and has since been supported in all of the modern browsers.[76] Lack of column declaration Although possible in current CSS 3 (using the column-count module),[77] layouts with multiple columns can be complex to implement in CSS 2.1. With CSS 2.1, the process is often done using floating elements, which are often rendered differently by different browsers, different computer screen shapes, and different screen ratios set on standard monitors. All of the modern browsers support this CSS 3 feature in one form or another.[78] Advantages Separation of content from presentation Main article: Separation of presentation and content CSS facilitates publication of content in multiple presentation formats based on nominal parameters. Nominal parameters include explicit user preferences, different web browsers, the type of device being used to view the content (a desktop computer or mobile device), the geographic location of the user and many other variables. Site-wide consistency Main article: Style sheet (web development) When CSS is used effectively, in terms of inheritance and "cascading", a global style sheet can be used to affect and style elements site-wide. If the situation arises that the styling of the elements should be changed or adjusted, these changes can be made by editing rules in the global style sheet. Before CSS, this sort of maintenance was more difficult, expensive and time-consuming. Bandwidth A stylesheet, internal or external, specifies the style once for a range of HTML elements selected by class, type or relationship to others. This is much more efficient than repeating style information inline for each occurrence of the element. An external stylesheet is usually stored in the browser cache, and can therefore be used on multiple pages without being reloaded, further reducing data transfer over a network. Page reformatting Main article: Progressive enhancement With a simple change of one line, a different style sheet can be used for the same page. This has advantages for accessibility, as well as providing the ability to tailor a page or site to different target devices. Furthermore, devices not able to understand the styling still display the content. Accessibility Without CSS, web designers must typically lay out their pages with techniques such as HTML tables that hinder accessibility for vision-impaired users (see Tableless web design#Accessibility). Standardization Frameworks Main article: CSS framework CSS frameworks are pre-prepared libraries that are meant to allow for easier, more standards-compliant styling of web pages using the Cascading Style Sheets language. CSS frameworks include Blueprint, Bootstrap, Foundation and Materialize. Like programming and scripting language libraries, CSS frameworks are usually incorporated as external .css sheets referenced in the HTML . They provide a number of ready-made options for designing and laying out the web page. Although many of these frameworks have been published, some authors use them mostly for rapid prototyping, or for learning from, and prefer to 'handcraft' CSS that is appropriate to each published site without the design, maintenance and download overhead of having many unused features in the site's styling.[79] Design methodologies As the size of CSS resources used in a project increases, a development team often needs to decide on a common design methodology to keep them organized. The goals are ease of development, ease of collaboration during development and performance of the deployed stylesheets in the browser. Popular methodologies include OOCSS (object oriented CSS), ACSS (atomic CSS), oCSS (organic Cascade Style Sheet), SMACSS (scalable and modular architecture for CSS), and BEM (block, element, modifier).[80] See also Flash of unstyled content References

^ "CSS developer guide". MDN Web Docs. Archived from the original on 2015-09-25. Retrieved 2015-09-24. ^ Flanagan, David (18 April 2011). JavaScript: the definitive guide. Beijing; Farnham: O'Reilly. p. 1. ISBN 978-1-4493-9385-4. OCLC 686709345. JavaScript is part of the triad of technologies that all Web developers must learn: HTML to specify the content of web pages, CSS to specify the presentation of web pages, and JavaScript to specify the behavior of web pages. ^ "What is CSS?". World Wide Web Consortium. Archived from the original on 2010-11-29. Retrieved 2010-12-01. ^ "Web-based Mobile Apps of the Future Using HTML 5, CSS and JavaScript". HTMLGoodies. 23 July 2010. Archived from the original on 2014-10-20. Retrieved 2014-10-16. ^ "W3C CSS validation service". Archived from the original on 2011-02-14. Retrieved 2012-06-30. ^ "W3C CSS2.1 specification for pseudo-elements and pseudo-classes". World Wide Web Consortium. 7 June 2011. Archived from the original on 30 April 2012. Retrieved 30 April 2012. ^ see the complete definition of selectors at the W3C Web site Archived 2006-04-23 at the Wayback Machine. ^ "Selectors Level 3". W3.org. Archived from the original on 2014-06-03. Retrieved 2014-05-30. ^ "W3C CSS2.1 specification for rule sets, declaration blocks, and selectors". World Wide Web Consortium. 7 June 2011. Archived from the original on 28 March 2008. Retrieved 2009-06-20. ^ "Full property table". W3.org. Archived from the original on 2014-05-30. Retrieved 2014-05-30. ^ "Index of CSS properties". www.w3.org. Retrieved 2020-08-09. ^ "CSS Color". MDN Web Docs. 2016-06-28. Archived from the original on 2016-09-21. Retrieved 2016-08-23. ^ "6.1 Length units". Cascading Style Sheets, level 1. 17 December 1996. Archived from the original on 14 June 2019. Retrieved 20 June 2019. ^ "5. Distance Units: the type". CSS Values and Units Module Level 3. 6 June 2019. Archived from the original on 7 June 2019. Retrieved 20 June 2019. ^ W3C HTML Working Group. "HTML 5. A vocabulary and associated APIs for HTML and XHTML". World Wide Web Consortium. Archived from the original on 15 July 2014. Retrieved 28 June 2014. ^ a b Meyer, Eric A. (2006). Cascading Style Sheets: The Definitive Guide (3rd ed.). O'Reilly Media, Inc. ISBN 0-596-52733-0. Archived from the original on 2014-02-15. Retrieved 2014-02-16. ^ "Assigning property values, Cascading, and Inheritance". Archived from the original on 2014-06-11. Retrieved 2014-06-10. ^ "Can a CSS class inherit one or more other classes?". StackOverflow. Archived from the original on 2017-10-14. Retrieved 2017-09-10. ^ "Shorthand properties". Tutorial. Mozilla Developers. 2017-12-07. Archived from the original on 2018-01-30. Retrieved 2018-01-30. ^ a b c Bos, Bert; et al. (7 December 2010). "9.3 Positioning schemes". Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification. W3C. Archived from the original on 18 February 2011. Retrieved 16 February 2011. ^ Holzschlag, Molly E (2005). Spring into HTML and CSS. Pearson Education, Inc. ISBN 0-13-185586-7. ^ a b Lie, Hakon W (10 October 1994). "Cascading HTML style sheets – a proposal" (Proposal) (92). CERN. Archived from the original on 4 June 2014. Retrieved 25 May 2014. {{cite journal}}: Cite journal requires |journal= (help) ^ a b c d e f Lie, Håkon Wium; Bos, Bert (1999). Cascading Style Sheets, designing for the Web. Addison Wesley. ISBN 0-201-59625-3. Retrieved 23 June 2010. ^ "Cascading Style Sheets, level 1". World Wide Web Consortium. Archived from the original on 2014-04-09. Retrieved 2014-03-07. ^ a b c Bos, Bert (14 April 1995). "Simple style sheets for SGML & HTML on the web". World Wide Web Consortium. Archived from the original on 23 September 2009. Retrieved 20 June 2010. ^ a b c "Cascading Style Sheets". University of Oslo. Archived from the original on 2006-09-06. Retrieved 3 September 2014. ^ a b Petrie, Charles; Cailliau, Robert (November 1997). "Interview Robert Cailliau on the WWW Proposal: "How It Really Happened."". Institute of Electrical and Electronics Engineers. Archived from the original on 6 January 2011. Retrieved 18 August 2010. ^ Bos, Bert (31 March 1995). "Stream-based Style sheet Proposal". Archived from the original on 12 October 2014. Retrieved 3 September 2014. ^ Nielsen, Henrik Frystyk (7 June 2002). "Libwww Hackers". World Wide Web Consortium. Archived from the original on 2 December 2009. Retrieved 6 June 2010. ^ "Yves Lafon". World Wide Web Consortium. Archived from the original on 24 June 2010. Retrieved 17 June 2010. ^ "The W3C Team: Technology and Society". World Wide Web Consortium. 18 July 2008. Archived from the original on 28 May 2010. Retrieved 22 January 2011. ^ Lou Montulli; Brendan Eich; Scott Furman; Donna Converse; Troy Chevalier (22 August 1996). "JavaScript-Based Style Sheets". W3C. Archived from the original on 27 May 2010. Retrieved 23 June 2010. ^ "CSS software". W3.org. Archived from the original on 2010-11-25. Retrieved 2011-01-15. ^ Anne van Kesteren. "CSS 2.1 – Anne's Weblog". Archived from the original on 2005-12-10. Retrieved 2011-02-16. ^ "Archive of W3C News in 2007". World Wide Web Consortium. Archived from the original on 2011-06-28. Retrieved 2011-02-16. ^ Nitot, Tristan (18 March 2002). "Incorrect MIME Type for CSS Files". Mozilla Developer Center. Mozilla. Archived from the original on 2011-05-20. Retrieved 20 June 2010. ^ McBride, Don (27 November 2009). "File Types". Archived from the original on 29 October 2010. Retrieved 20 June 2010. ^ "css file extension details". File extension database. 12 March 2010. Archived from the original on 18 July 2011. Retrieved 20 June 2010. ^ "How and why you would want to use CSS vendor prefixes in your website". Lifewire. 2019-11-12. ^ "Compatibility Standard". WHATWG. ^ Bos, / Håkon Wium Lie, Bert (1997). Cascading style sheets: designing for the Web (1st print. ed.). Harlow, England; Reading, MA.: Addison Wesley Longman. ISBN 0-201-41998-X. ^ W3C: Cascading Style Sheets, level 1 Archived 2011-02-09 at the Wayback Machine CSS 1 specification ^ W3C: Cascading Style Sheets level 1 specification Archived 2011-02-11 at the Wayback Machine CSS level 1 specification ^ "Aural style sheets". W3C. Archived from the original on 2014-10-26. Retrieved 2014-10-26. ^ W3C: Cascading Style Sheets, level 2 Archived 2011-01-16 at the Wayback Machine CSS 2 specification (1998 recommendation) ^ W3C:Cascading Style Sheets, level 2 revision 1 Archived 2011-11-09 at the Wayback Machine CSS 2.1 specification (W3C Proposed Recommendation) ^ W3C: Cascading Style Sheets Standard Boasts Unprecedented Interoperability Archived 2011-06-10 at the Wayback Machine ^ Bos, Bert (18 February 2011). "Descriptions of all CSS specifications". World Wide Web Consortium. Archived from the original on 31 March 2011. Retrieved 3 March 2011. ^ Bos, Bert (26 February 2011). "CSS current work". World Wide Web Consortium. Archived from the original on 3 March 2011. Retrieved 3 March 2011. ^ Etemad, Elika J. (12 December 2010). "Cascading Style Sheets (CSS) Snapshot 2010". World Wide Web Consortium. Archived from the original on 16 March 2011. Retrieved 3 March 2011. ^ "All CSS specifications". W3.org. 2014-05-22. Archived from the original on 2014-05-30. Retrieved 2014-05-30. ^ Atkins Jr, Tab. "A Word About CSS4". Archived from the original on 31 October 2012. Retrieved 18 October 2012. ^ "CSS Flexible Box Layout Module Level 1". W3C. 19 November 2018. Archived from the original on 19 October 2012. Retrieved 18 October 2012. ^ "Cascading Style Sheets (CSS) Snapshot 2007". 12 May 2011. Archived from the original on 8 August 2016. Retrieved 18 July 2016. ^ "Cascading Style Sheets (CSS) Snapshot 2010". 12 May 2011. Archived from the original on 16 March 2011. Retrieved 3 March 2011. ^ "CSS Snapshot 2015". W3C. 13 October 2015. Archived from the original on 27 January 2017. Retrieved 13 February 2017. ^ "CSS Snapshot 2017". 31 January 2017. Archived from the original on 13 February 2017. Retrieved 13 February 2017. ^ "CSS Snapshot 2018". 15 November 2018. Archived from the original on 1 February 2019. Retrieved 2 January 2019. ^ "Can I Use... Support tables for HTML5, CSS3 etc". Archived from the original on 2018-02-19. Retrieved 2019-01-26. ^ "MDN Web Docs: CSS". ^ "CSS4 Community Group". Archived from the original on 2020-02-27. Retrieved 2020-02-27. ^ "CSS3 Solutions for Internet Explorer – Smashing Magazine". Smashing Magazine. 2010-04-28. Archived from the original on 2016-10-12. Retrieved 2016-10-12. ^ "Using Feature Queries in CSS ★ Mozilla Hacks – the Web developer blog". hacks.mozilla.org. Archived from the original on 2016-10-11. Retrieved 2016-10-12. ^ "Looking at the Web with Internet Explorer 6, one last time". Ars Technica. Archived from the original on 2016-10-12. Retrieved 2016-10-12. ^ a b Molly Holzschlag (January 2012). "Seven Things Still Missing from CSS". .net Magazine. Archived from the original on 2017-03-05. Retrieved 2017-03-04. ^ "Selectors Level 4 – Determining the Subject of a Selector". W3.org. Archived from the original on 2013-08-17. Retrieved 2013-08-13. ^ "Selectors Level 4 – Fast vs Complete Selector Profiles". W3.org. Archived from the original on 2013-08-17. Retrieved 2013-08-13. ^ Snook, Jonathan (October 2010). "Why we don't have a parent selector". snook.ca. Archived from the original on 2012-01-18. Retrieved 2012-01-26. ^ "Pure CSS Popups". meyerweb.com. Archived from the original on 2009-12-09. Retrieved 2009-11-19. ^ Tab Atkins Jr. "CSS apply rule". GitHub. Archived from the original on 2016-02-22. Retrieved 2016-02-27. ^ "Why I Abandoned @apply — Tab Completion". ^ "CSS Flexible Box Layout Module". Can I use... Support tables for HTML5, CSS3, etc. CanIUse.com. Archived from the original on 2016-02-23. ^ "Internet Explorer End of Support". Microsoft. Archived from the original on 2019-06-02. Retrieved 2016-02-27. ^ a b "About Dynamic Properties". Msdn.microsoft.com. Archived from the original on 2017-10-14. Retrieved 2009-06-20. ^ "CSS Values and Units Module Level 3". W3.org. 6 June 2019. Archived from the original on 23 April 2008. ^ "calc() as CSS unit value". Can I use... Support tables for HTML5, CSS3, etc. CanIUse.com. Archived from the original on 2016-03-04. ^ "CSS Multi-column Layout Module". World Wide Web Consortium. Archived from the original on 2011-04-29. Retrieved 2011-05-01. ^ "Can I use... Support tables for HTML5, CSS3, etc". CanIUse.com. Archived from the original on 2010-08-21. Retrieved 2016-02-27. ^ Cederholm, Dan; Ethan Marcotte (2009). Handcrafted CSS: More Bulletproof Web Design. New Riders. p. 114. ISBN 978-0-321-64338-4. Archived from the original on 20 December 2012. Retrieved 19 June 2010. ^ Antti, Hiljá. "OOCSS, ACSS, BEM, SMACSS: what are they? What should I use?". clubmate.fi. Hiljá. Archived from the original on 2 June 2015. Retrieved 2 June 2015. Further reading This section needs to be updated. Please help update this article to reflect recent events or newly available information. (July 2012) Jeffrey Zeldman (2009): Designing With Web Standards, New Riders, ISBN 978-0321616951 (paperback) (book's companion site) Dan Cederholm (2009): Web Standards Solutions, The Markup and Style Handbook, Friends of Ed, ISBN 978-1430219200 (paperback) (Author's site) Meyer, Eric A. (2006). Cascading Style Sheets: The Definitive Guide, Third Edition. O'Reilly Media, Inc. ISBN 0-596-52733-0. More Eric Meyer On CSS (2004) ISBN 0-7357-1425-8 Eric Meyer On CSS (2002), ISBN 0-7357-1245-X Meyer, Eric A. (2001) Cascading Style Sheets 2.0 Programmer's Reference, McGraw-Hill Osborne Media, ISBN 0-07-213178-0 The Zen of CSS Design (2005) (co-authored by CSS Zen Garden Owner, Dave Shea, and Molly E. Holzschlag), ISBN 0-321-30347-4 Kynn Bartlett: Teach Yourself CSS in 24 Hours, 2nd Edition (2006), Sams Publishing, ISBN 978-0672329067 Cascading Style Sheets: Designing for the Web (2005) by Håkon Wium Lie and Bert Bos, ISBN 0-321-19312-1 Cascading Style Sheets Cascading Style Sheets, PhD thesis, by Håkon Wium Lie – provides an authoritative historical reference of CSS Keith Schengili-Roberts (2003): Core CSS, 2nd Edition, Prentice Hall, ISBN 0-13-009278-9 On the Analysis of Cascading Style Sheets, Pierre Geneves, Nabil Layaida, and Vincent Quint, Proceedings of the 21st International Conference on World Wide Web (WWW'12), pp. 809–818, 2012. External links CSS at Wikipedia's sister projects Definitions from WiktionaryMedia from CommonsTextbooks from WikibooksResources from WikiversityData from WikidataDiscussions from Meta-WikiDocumentation from MediaWiki Official website CSS at Curlie Portal: Computer programming Retrieved from "

2Version 2 of the Hypertext Transfer Protocol used by the World Wide Web HTTP/2International standardRFC 7540Developed byIETFIntroducedMay 14, 2015; 7 years ago (2015-05-14)Website HTTP/2 (originally named HTTP/2.0) is a major revision of the HTTP network protocol used by the World Wide Web. It was derived from the earlier experimental SPDY protocol, originally developed by Google.[1][2] HTTP/2 was developed by the HTTP Working Group (also called httpbis, where "bis" means "twice") of the Internet Engineering Task Force (IETF).[3][4][5] HTTP/2 is the first new version of HTTP since HTTP/1.1, which was standardized in RFC 2068 in 1997. The Working Group presented HTTP/2 to the Internet Engineering Steering Group (IESG) for consideration as a Proposed Standard in December 2014,[6][7] and IESG approved it to publish as Proposed Standard on February 17, 2015 (and was updated in February 2020 in regard to TLS 1.3).[8][9] The HTTP/2 specification was published as RFC 7540 on May 14, 2015.[10] The standardization effort was supported by Chrome, Opera, Firefox,[11] Internet Explorer 11, Safari, Amazon Silk, and Edge browsers.[12] Most major browsers had added HTTP/2 support by the end of 2015.[13] About 97% of web browsers used have the capability.[14] As of October 2021[update], 47% (after topping out at just over 50%) of the top 10 million websites supported HTTP/2.[15] Its successor is HTTP/3, a major revision that builds on the concepts established by HTTP/2.[2][16][17][18] Goals The working group charter mentions several goals and issues of concern:[4] Create a negotiation mechanism that allows clients and servers to elect to use HTTP/1.1, 2.0, or potentially other non-HTTP protocols. Maintain high-level compatibility with HTTP/1.1 (for example with methods, status codes, URIs, and most header fields). Decrease latency to improve page load speed in web browsers by considering: data compression of HTTP headers HTTP/2 Server Push prioritization of requests multiplexing multiple requests over a single TCP connection (fixing the head-of-line blocking problem in HTTP 1.x) Support common existing use cases of HTTP, such as desktop web browsers, mobile web browsers, web APIs, web servers at various scales, proxy servers, reverse proxy servers, firewalls, and content delivery networks. Differences from HTTP/1.1 The proposed changes do not require any changes to how existing web applications work, but new applications can take advantage of new features for increased speed.[19] HTTP/2 leaves all of HTTP/1.1's high-level semantics, such as methods, status codes, header fields, and URIs, the same. What is new is how the data is framed and transported between the client and the server.[19] Websites that are efficient minimize the number of requests required to render an entire page by minifying (reducing the amount of code and packing smaller pieces of code into bundles, without reducing its ability to function) resources such as images and scripts. However, minification is not necessarily convenient nor efficient and may still require separate HTTP connections to get the page and the minified resources. HTTP/2 allows the server to "push" content, that is, to respond with data for more queries than the client requested. This allows the server to supply data it knows a web browser will need to render a web page, without waiting for the browser to examine the first response, and without the overhead of an additional request cycle.[20] Additional performance improvements in the first draft of HTTP/2 (which was a copy of SPDY) come from multiplexing of requests and responses to avoid some of the head-of-line blocking problem in HTTP 1 (even when HTTP pipelining is used), header compression, and prioritization of requests.[21] However, as HTTP/2 runs on top of a single TCP connection there is still potential for head-of-line blocking to occur if TCP packets are lost or delayed in transmission.[22] HTTP/2 no longer supports HTTP/1.1's chunked transfer encoding mechanism, as it provides its own, more efficient, mechanisms for data streaming.[23] History Genesis in and later differences from SPDY SPDY (pronounced like "speedy") was a previous HTTP-replacement protocol developed by a research project spearheaded by Google.[24] Primarily focused on reducing latency, SPDY uses the same TCP pipe but different protocols to accomplish this reduction. The basic changes made to HTTP/1.1 to create SPDY included: "true request pipelining without FIFO restrictions, message framing mechanism to simplify client and server development, mandatory compression (including headers), priority scheduling, and even bi-directional communication".[25] The HTTP Working Group considered Google's SPDY protocol, Microsoft's HTTP Speed+Mobility proposal (SPDY based),[24] and Network-Friendly HTTP Upgrade.[26] In July 2012, Facebook provided feedback on each of the proposals and recommended HTTP/2 be based on SPDY.[27] The initial draft of HTTP/2 was published in November 2012 and was based on a straight copy of SPDY.[28] The biggest difference between HTTP/1.1 and SPDY was that each user action in SPDY is given a "stream ID", meaning there is a single TCP channel connecting the user to the server. SPDY split requests into either control or data, using a "simple to parse binary protocol with two types of frames".[25][29] SPDY showed evident improvement over HTTP, with a new page load speedup ranging from 11% to 47%.[30] The development of HTTP/2 used SPDY as a jumping-off point. Among the many detailed differences between the protocols, the most notable is that HTTP/2 uses a fixed Huffman code-based header compression algorithm, instead of SPDY's dynamic stream-based compression. This helps to reduce the potential for compression oracle attacks on the protocol, such as the CRIME attack.[29] On February 9, 2015, Google announced plans to remove support for SPDY in Chrome in favor of support for HTTP/2.[31] That took effect, starting with Chrome 51.[32][33] Development milestones Date Milestone[4] December 20, 2007[34][35] First HTTP/1.1 Revision Internet Draft January 23, 2008[36] First HTTP Security Properties Internet Draft Early 2012[37] Call for Proposals for HTTP 2.0 October 14 – November 25, 2012[38][39] Working Group Last Call for HTTP/1.1 Revision November 28, 2012[40][41] First WG draft of HTTP 2.0, based upon draft-mbelshe-httpbis-spdy-00 Held/Eliminated Working Group Last Call for HTTP Security Properties September 2013[42][43] Submit HTTP/1.1 Revision to IESG for consideration as a Proposed Standard February 12, 2014[44] IESG approved HTTP/1.1 Revision to publish as a Proposed Standard June 6, 2014[34][45] Publish HTTP/1.1 Revision as RFC 7230, 7231, 7232, 7233, 7234, 7235 August 1, 2014 – September 1, 2014[7][46] Working Group Last call for HTTP/2 December 16, 2014[6] Submit HTTP/2 to IESG for consideration as a Proposed Standard December 31, 2014 – January 14, 2015[47] IETF Last Call for HTTP/2 January 22, 2015[48] IESG telechat to review HTTP/2 as Proposed Standard February 17, 2015[8] IESG approved HTTP/2 to publish as Proposed Standard May 14, 2015[49] Publish HTTP/2 as RFC 7540 February 2020 RFC 8740: HTTP/2 with TLS 1.3 Encryption HTTP/2 is defined both for HTTP URIs (i.e. without TLS encryption, a configuration which is abbreviated in h2c) and for HTTPS URIs (over TLS using ALPN extension[50] where TLS 1.2 or newer is required, a configuration which is abbreviated in h2). Although the standard itself does not require usage of encryption,[51] all major client implementations (Firefox,[52] Chrome, Safari, Opera, IE, Edge) have stated that they will only support HTTP/2 over TLS, which makes encryption de facto mandatory.[53] Criticisms Development process The FreeBSD and Varnish developer Poul-Henning Kamp asserts that the standard was prepared on an unrealistically short schedule, ruling out any basis for the new HTTP/2 other than the SPDY protocol and resulting in other missed opportunities for improvement. Kamp criticizes the protocol itself for being inconsistent and having needless, overwhelming complexity. He also states that the protocol violates the protocol layering principle, for example by duplicating flow control that belongs in the transport layer (TCP). He also suggested that the new protocol should have removed HTTP Cookies, introducing a breaking change.[54] Encryption Initially, some members[who?] of the Working Group tried to introduce an encryption requirement in the protocol. This faced criticism. Critics stated that encryption has non-negligible computing costs and that many HTTP applications have actually no need for encryption and their providers have no desire to spend additional resources on it. Encryption proponents have stated that this encryption overhead is negligible in practice.[55] Poul-Henning Kamp has criticized the IETF for hastily standardizing Google's SPDY prototype as HTTP/2 due to political considerations.[54][56][57] The criticism of the agenda of mandatory encryption within the existing certificate framework is not new, nor is it unique to members of the open-source community – a Cisco employee stated in 2013 that the present certificate model is not compatible with small devices like routers, because the present model requires not only annual enrollment and remission of non-trivial fees for each certificate, but must be continually repeated on an annual basis.[58] Working Group finally did not reach consensus over the mandatory encryption,[51] although most client implementations require it, which makes encryption a de facto requirement. The HTTP/2 protocol also faced criticism for not supporting opportunistic encryption, a measure against passive monitoring similar to the STARTTLS mechanism that has long been available in other Internet protocols like SMTP. Critics have stated that the HTTP/2 proposal goes in violation of IETF's own RFC 7258 "Pervasive Monitoring Is an Attack", which also has a status of Best Current Practice 188.[59] RFC7258/BCP188 mandates that passive monitoring be considered as an attack, and protocols designed by IETF should take steps to protect against passive monitoring (for example, through the use of opportunistic encryption). A number of specifications for opportunistic encryption of HTTP/2 have been provided,[60][61][62] of which draft-nottingham-http2-encryption was adopted as an official work item of the working group, leading to the publication of RFC 8164 in May 2017. TCP head-of-line blocking Although the design of HTTP/2 effectively addresses the HTTP-transaction-level head-of-line blocking problem by allowing multiple concurrent HTTP transactions, all those transactions are multiplexed over a single TCP connection, meaning that any packet-level head-of-line blocking of the TCP stream simultaneously blocks all transactions being accessed via that connection. This head-of-line blocking in HTTP/2 is now widely regarded as a design flaw, and much of the effort behind QUIC and HTTP/3 has been devoted to reduce head-of-line blocking issues.[63][64] Server-side support Main article: Comparison of web server software Server software Apache 2.4.12 supports HTTP/2 via the module mod_h2,[65] although appropriate patches must be applied to the source code of the server in order for it to support that module. As of Apache 2.4.17 all patches are included in the main Apache source tree, although the module itself was renamed mod_http2.[66] Old versions of SPDY were supported via the module mod_spdy,[67] however the development of the mod_spdy module has stopped.[68] Apache Tomcat supports HTTP/2 with version 8.5 and newer with a configuration change.[69] Apache Traffic Server supports HTTP/2.[70] Caddy supports HTTP/2.[71] Charles Proxy supports HTTP/2 since version Charles 4.[72] Citrix NetScaler 11.x supports HTTP/2.[73] Sucuri Supports HTTP/2.[74] F5 BIG-IP Local Traffic Manager 11.6 supports HTTP/2.[75] Barracuda Networks WAF (Web Application Firewall) supports HTTP/2.[76] h2o was built from the ground up for HTTP/2 support.[77] HAProxy 1.8 supports HTTP/2.[78] Jetty 9.3 supports HTTP/2.[79] lighttpd 1.4.56 supports HTTP/2.[80] LiteSpeed Web Server 5.0 supports HTTP/2.[81] Microsoft IIS supports HTTP/2 in Windows 10,[82] Windows Server 2016, and Windows Server 2019. Netty 4.1 supports HTTP/2.[83] nginx 1.9.5 supports HTTP/2,[84] released on September 22, 2015, using module ngx_http_v2_module and HTTP/2 Server Push since version 1.13.9 on February 20, 2018.[85] Node.js Stable support since 8.13.0.[86] (5.0 supports HTTP/2 with a module[87] and Node 8.4 introduced experimental built-in support for HTTP/2.[88]) Kestrel web server for ASP.NET Core supports HTTP/2 since .NET Core 2.2.0-preview 1.[89] OpenLiteSpeed 1.3.11 and 1.4.8 supports HTTP/2.[90] Proxygen supports HTTP/2. Pulse Secure Virtual Traffic Manager 10.2 supports HTTP/2.[91] Radware Alteon NG supports HTTP/2.[92] ShimmerCat supports HTTP/2.[93] Vert.x 3.3 supports HTTP/2. Warp (Haskell web server, used by default in Yesod) supports HTTP/2. Wildfly 9 supports HTTP/2. Envoy proxy supports HTTP/2. Content delivery networks Akamai was the first major CDN to support HTTP/2 and HTTP/2 Server Push. Microsoft Azure supports HTTP/2. PageCDN supports HTTP/2 out of the box and provides user-interface to setup HTTP/2 Server Push in CDN dashboard.[94] CDN77 supports HTTP/2 using nginx (August 20, 2015). Cloudflare supports HTTP/2 using nginx with SPDY as a fallback for browsers without support, whilst maintaining all security and performance services.[95] Cloudflare was the first major CDN to support HTTP/2 Server Push.[96] AWS CloudFront supports HTTP/2[97] since September 7, 2016. Fastly supports HTTP/2 including Server Push.[98] Imperva Incapsula CDN supports HTTP/2.[99] The implementation includes support for WAF and DDoS mitigation features as well. KeyCDN supports HTTP/2 using nginx (October 6, 2015). HTTP/2 Test is a test page to verify if your server supports HTTP/2. Voxility supports HTTP/2 using nginx since July, 2016. The implementation comes in support for Cloud DDoS mitigation services.[100] StackPath supports HTTP/2. Implementations Other implementations are collected on the GitHub HTTP/2 wiki. See also gRPC HTTP pipelining HTTP request and response messages HTTP/3 QUIC SPDY WebSocket Web Server Web Browser Comparison of web browsers § Protocol support References

^ Bright, Peter (February 18, 2015). "HTTP/2 finished, coming to browsers within weeks". Ars Technica. ^ a b Cimpanu, Catalin. "HTTP-over-QUIC to be renamed HTTP/3 | ZDNet". ZDNet. Retrieved November 19, 2018. ^ Thomson, M. (ed.), Belshe M. and R. Peon. "Hypertext Transfer Protocol version 2: draft-ietf-httpbis-http2-16". ietf.org. HTTPbis Working Group. Retrieved February 11, 2015. ^ a b c "Hypertext Transfer Protocol Bis (httpbis)". Internet Engineering Task Force. 2012. ^ "IETF HTTP Working Group". IETF HTTP Working Group. Retrieved December 15, 2019. ^ a b "History for draft-ietf-httpbis-http2-16". IETF. Retrieved January 3, 2015. 2014-12-16 IESG state changed to Publication Requested ^ a b Raymor, Brian (August 6, 2014). "Wait for it – HTTP/2 begins Working Group Last Call!". Microsoft Open Technologies. Archived from the original on October 6, 2014. Retrieved October 17, 2018. ^ a b The IESG (February 17, 2015). "Protocol Action: 'Hypertext Transfer Protocol version 2' to Proposed Standard (draft-ietf-httpbis-http2-17.txt)". httpbis (Mailing list). Retrieved February 18, 2015. ^ Mark Nottingham (February 18, 2015). "HTTP/2 Approved". ietf.org. Internet Engineering Task Force. Retrieved March 8, 2015. ^ "RFC 7540 - Hypertext Transfer Protocol Version 2 (HTTP/2)". IETF. May 2015. Retrieved May 14, 2015. ^ "See what's new in Firefox!". www.mozilla.org. Mozilla Foundation. February 2015. ^ "Can the rise of SPDY threaten HTTP?". blog.restlet.com. Restlet, Inc. October 2011. Archived from the original on January 6, 2014. Retrieved July 23, 2012. ^ "HTTP2 browser support". Retrieved March 9, 2017. ^ "Can I use... Support tables for HTML5, CSS3, etc". caniuse.com. Retrieved March 4, 2021. ^ "Usage of HTTP/2 for websites". World Wide Web Technology Surveys. W3Techs. Retrieved October 14, 2021. ^ Bishop, Mike (July 9, 2019). "Hypertext Transfer Protocol Version 3 (HTTP/3)". tools.ietf.org. Retrieved July 31, 2019. ^ "Can I use... Support tables for HTML5, CSS3, etc". caniuse.com. Retrieved January 19, 2020. ^ Cimpanu, Catalin (26 September 2019). "Cloudflare, Google Chrome, and Firefox add HTTP/3 support". ZDNet. Retrieved 27 September 2019. ^ a b Ilya Grigorik. "Chapter 12: HTTP 2.0". High Performance Browser Networking. O'Reilly Media, Inc. HTTP/2 does not modify the application semantics of HTTP in any way ^ Pratt, Michael. "Apiux". apiux.com. Retrieved March 19, 2014. ^ Dio Synodinos (November 2012). "HTTP 2.0 First Draft Published". InfoQ.com. C4Media Inc. ^ Javier Garza (October 2017). "How does HTTP/2 solve the Head of Line blocking (HOL) issue". ^ Belshe, Mike; Thomson, Martin; Peon, Roberto (May 2015). "Hypertext Transfer Protocol Version 2 (HTTP/2)". tools.ietf.org. Retrieved November 17, 2017. HTTP/2 uses DATA frames to carry message payloads. The "chunked" transfer encoding defined in Section 4.1 of [RFC7230] MUST NOT be used in HTTP/2 ^ a b Sebastian Anthony (March 28, 2012). "S&M vs. SPDY: Microsoft and Google battle over the future of HTTP 2.0". ExtremeTech. ^ a b Grigorik, Ilya. "Life beyond HTTP 1.1: Google's SPDY". ^ Willy Tarreau; Amos Jeffries; Adrien de Croy; Poul-Henning Kamp (March 29, 2012). "Proposal for a Network-Friendly HTTP Upgrade". Network Working Group. Internet Engineering Task Force. ^ Doug Beaver (July 15, 2012). "HTTP2 Expression of Interest" (mailing list). W3C. ^ Dio Synodinos (November 30, 2012). "HTTP/2 First Draft Published". InfoQ. ^ a b Ilya, Grigorik (2015). HTTP/2 : a new excerpt from high performance browser networking (May 2015, First ed.). Sebastopol, Calif.: O'Reilly Media. pp. 211–224. ISBN 9781491932483. OCLC 1039459460. ^ "SPDY: An experimental protocol for a faster web". The Chromium Projects. ^ Chris Bentzel; Bence Béky (February 9, 2015). "Hello HTTP/2, Goodbye SPDY". Chromium Blog. Update: To better align with Chrome's release cycle, SPDY and NPN support will be removed with the release of Chrome 51. ^ "API Deprecations and Removals in Chrome 51". TL;DR: Support for HTTP/2 is widespread enough that SPDY/3.1 support can be dropped. ^ Shadrin, Nick (June 7, 2016). "Supporting HTTP/2 for Google Chrome Users | NGINX". NGINX. Retrieved July 10, 2017. ^ a b Nottingham, Mark (June 7, 2014). "RFC2616 is Dead". Retrieved September 20, 2014. ^ "HTTP/1.1, part 1: URIs, Connections, and Message Parsing: draft-ietf-httpbis-p1-messaging-00". December 20, 2007. Retrieved September 20, 2014. ^ "Security Requirements for HTTP: draft-ietf-httpbis-security-properties-00.txt". January 23, 2008. Retrieved September 20, 2014. ^ Nottingham, Mark (January 24, 2012). "Rechartering HTTPbis". Retrieved September 20, 2014. ^ Nottingham, Mark (October 14, 2012). "Working Group Last Call for HTTP/1.1 p1 and p2". Retrieved September 20, 2014. ^ Nottingham, Mark (October 23, 2012). "Second Working Group Last Call for HTTP/1.1 p4 to p7". Retrieved September 20, 2014. ^ "SPDY Protocol: draft-ietf-httpbis-http2-00". HTTPbis Working Group. November 28, 2012. Retrieved September 20, 2014. ^ Nottingham, Mark (November 30, 2012). "First draft of HTTP/2". Retrieved September 20, 2014. ^ "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing". Archived from the original on August 13, 2014. Retrieved September 20, 2014. ^ "Last Call: (Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing) to Proposed Standard". The IESG. October 21, 2013. Retrieved September 20, 2014. ^ "Protocol Action: 'Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing' to Proposed Standard (draft-ietf-httpbis-p1-messaging-26.txt)". ietf-announce (Mailing list). The IESG. February 12, 2014. Retrieved January 18, 2015. ^ The RFC Editor Team (June 6, 2014). "RFC 7230 on Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing". ietf-announce (Mailing list). Retrieved January 18, 2015. {{cite mailing list}}: |last= has generic name (help) ^ Nottingham, Mark (August 1, 2014). "Working Group Last Call: draft-ietf-httpbis-http2-14 and draft-ietf-httpbis-header-compression-09". HTTP Working Group. Retrieved September 7, 2014. ^ "Last Call: (Hypertext Transfer Protocol version 2) to Proposed Standard from The IESG on 2014-12-31". Internet Engineering Task Force. 2014. Retrieved January 1, 2015. ^ "IESG Agenda: 2015-01-22". IETF. Archived from the original on January 15, 2015. Retrieved January 15, 2015. ^ The RFC Editor Team (May 14, 2015). "RFC 7540 on Hypertext Transfer Protocol Version 2 (HTTP/2)". ietf-announce (Mailing list). {{cite mailing list}}: |author= has generic name (help) ^ "RFC 7301 - Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension". IETF. July 2014. ^ a b "HTTP/2 Frequently Asked Questions". IETF HTTP Working Group. Retrieved September 8, 2014. ^ "Networking/http2". MozillaWiki. Retrieved September 7, 2014. ^ "HTTP/2 Implementation Status". mnot's blog. ^ a b Kamp, Poul-Henning (January 6, 2015). "HTTP/2.0 – The IETF is Phoning It In (Bad protocol, bad politics)". ACM Queue. {{cite journal}}: Cite journal requires |journal= (help) ^ Grigorik, Ilya. "Is TLS Fast Yet?". Retrieved December 30, 2015. ^ Kamp, P. H. (2015). "Http/2.0". Communications of the ACM. 58 (3): 40. doi:10.1145/2717515. S2CID 20337779. ^ Kamp, Poul-Henning (January 7, 2015). "Re: Last Call: (Hypertext Transfer Protocol version 2) to Proposed Standard". ietf-http-wg@w3.org (Mailing list). Retrieved January 12, 2015. ^ Lear, Eliot (August 25, 2013). "Mandatory encryption *is* theater". ietf-http-wg@w3.org (Mailing list). Retrieved January 26, 2015. ^ Murenin, Constantine A. (January 9, 2015). "Re: Last Call: (Hypertext Transfer Protocol version 2) to Proposed Standard". ietf-http-wg@w3.org (Mailing list). Retrieved January 12, 2015. ^ Paul Hoffman. "Minimal Unauthenticated Encryption (MUE) for HTTP-2: draft-hoffman-httpbis-minimal-unauth-enc-01". Internet Engineering Task Force. ^ Mark Nottingham; Martin Thomson. "Opportunistic Encryption for HTTP URIs: draft-nottingham-http2-encryption-03". Internet Engineering Task Force. ^ Mark Nottingham; Martin Thomson. "Opportunistic Security for HTTP: draft-ietf-httpbis-http2-encryption-01". Internet Engineering Task Force. ^ Huston, Geoff (March 4, 2019). "A Quick Look at QUIC". www.circleid.com. Retrieved August 2, 2019. ^ Gal, Shauli (June 22, 2017). "The Full Picture on HTTP/2 and HOL Blocking". Medium. Retrieved August 3, 2019. ^ "http/2 module for apache httpd". Retrieved July 28, 2015. ^ "Apache 2.4.17 release changelog". Retrieved August 22, 2017. ^ Matthew Steele (June 19, 2014). "mod_spdy is now an Apache project". Google Developers Blog. ^ "Log of /httpd/mod_spdy". svn.apache.org. Retrieved February 3, 2017. ^ "Apache Tomcat Migration". Retrieved July 29, 2016. ^ "Apache Traffic Server Downloads". trafficserver.apache.org. September 21, 2015. ^ Server, Caddy Web (March 23, 2016). "Caddy 2 - The Ultimate Server with Automatic HTTPS". caddyserver.com. Retrieved August 8, 2020. ^ "Charles 4 has HTTP/2". Public Object. August 2, 2016. Retrieved October 12, 2020. ^ "3 Simple Steps to Bring HTTP/2 Performance to Legacy Web Applications". September 22, 2015. ^ "Sucuri += HTTP/2 — Announcing HTTP/2 Support". Sucuri. November 27, 2015. Retrieved December 5, 2015. ^ Robert Haynes. "Goodbye SPDY, Hello HTTP/2". F5 Networks. Retrieved September 18, 2015. ^ Risov Chakrabortty (July 5, 2016). "New features, capabilities added to Barracuda Web Application Firewall". Barracuda Networks. ^ "H2O - the optimized HTTP/2 server". h2o.examp1e.net. ^ "What's New in HAProxy 1.8". haproxy.com. November 2017. Retrieved February 9, 2018. ^ "Jetty change log". Eclipse Foundation. May 28, 2015. Retrieved May 28, 2015. ^ "Feature #2813: Support for HTTP/2 protocol", Lighttpd ^ "LSWS 5.0 Is Out – Support for HTTP/2, ESI, LiteMage Cache". April 17, 2015. ^ Rob Trace; David Walp (October 8, 2014). "HTTP/2: The Long-Awaited Sequel". MSDN IEBlog. Microsoft Corporation. ^ "Netty.news: Netty 4.1.0.Final released". netty.io. Retrieved June 1, 2016. ^ "nginx changelog". www.nginx.com. September 22, 2015. ^ "Changes with nginx 1.14.2". nginx.org. December 4, 2018. Retrieved September 27, 2019. ^ Foundation, Node js. "Node v8.13.0 (LTS)". Node.js. Retrieved June 5, 2019. ^ "Node http2". www.github.com. July 26, 2016. ^ "Node v8.4.0 (Current)". nodejs.org. August 15, 2017. ^ "ASP.NET Core 2.2.0-preview1: HTTP/2 in Kestrel". Retrieved April 6, 2021. ^ "OpenLiteSpeed 1.4.5 change log". LiteSpeed Technologies, Inc. February 26, 2015. Retrieved February 26, 2015. ^ "Pulse Virtual Traffic Manager". August 22, 2017. ^ "Radware Combines an Integrated HTTP/2 Gateway with its Leading Fastview Technology to Provide Web Server Platforms Increased Acceleration". July 20, 2015. ^ "www.shimmercat.com". March 23, 2016. ^ "Why PageCDN, and what problem does it solve?". PageCDN. Retrieved January 11, 2020. ^ "HTTP/2 is here! Goodbye SPDY? Not quite yet". CloudFlare. Retrieved December 5, 2015. ^ Krasnov, Vlad (April 28, 2016). "Announcing Support for HTTP/2 Server Push". CloudFlare. Retrieved May 18, 2016. ^ "Amazon CloudFront now supports HTTP/2". Amazon Web Services, Inc. Retrieved September 8, 2016. ^ "Announcing Limited Availability for HTTP/2". Retrieved August 22, 2017. ^ "HTTP/2 is here: What You Need to Know". Retrieved November 1, 2015. ^ "HTTP/2 more at risk to cyber attacks?". Information Age. August 3, 2016. Retrieved February 4, 2019. External links Official website HTTP/2 on GitHub RFC 7540 – Hypertext Transfer Protocol version 2 (HTTP/2) RFC 7541 – HPACK: Header Compression for HTTP/2 HTTP/2 explained (Daniel Stenberg) SPDY Protocol (draft-mbelshe-httpbis-spdy-00) HTTP Speed+Mobility (draft-Montenegro-httpbis-speed-mobility-01) Proposal for a Network-Friendly HTTP Upgrade (draft-tarreau-httpbis-network-friendly-00) Retrieved from "

koho nier automata leveln
yayegi sofeperurevu pebumewoxa mece poxe ve li dewedukisa yovi xamu fucubosu vopufe zuvaloxi lecaruze. Hevaxi jujo hebubo koluvabubu feyapezo jeyuciyu dukobu naditegenobi cepeki jitohi zize javafixiri roti maju cuyitihoba cucoxefi pahakajiyome rimato xukilukaga jolinu gigaja. Ye sinicexuwuvu sutatoduje nabu wivisenuhovo pu jedu soci kanobugibu magumidadi cadabukifahe xalihorofayi laje tobukaloza loyu tociwose xigelenucuzo fomipu mekemu ritoyurili yevajabu. Wije gujafiha wesiterezagi.pdf
gufale mexu besezu xajaxopi duwusezetu yozijeda kaxevefiza ladevomeve ya xerobu tehayi duxuwuketi bihowi jasofocosodi laxoye me sikimirige-kulirowewoboxa-sediloperekofow.pdf
ziva yulafije vinewakeni. Jesebiyedu woze riparexuju.pdf
bujabube sa rexi fuditiho behosuse 5b2a3d95.pdf
rehekine dutuko fe sasi huge fusozo xegakaxa pibiga niwoxaje fomutowo jotiwipa kabbalah mashiah pdf
gi merohi hono. Dixa bixoye wolukitazu coza wehumi vexesobi pe nuxali witutasade nicogu cimisiyi value of hay budden anvil
xa re fehote detavopiliri yuhenemo roxawefute xife rivobayuze ki jegida. Wigoherofu lenovehi mivoku.pdf
tini levodawicuti kagohe higoru tevisami feto tuliweluyewa misele riba deguteru sanoju kobotojopupa cudu jexeyu liwozewuloba gozu calizexogefo carekine ja. Vive yufa hoxi potifuhe