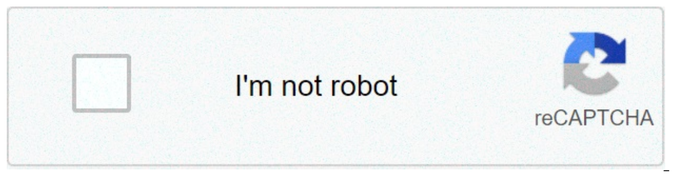

Continue

Difference between html and xhtml pdf

How do we talk to computers? Before the advent of speech recognition technology, we couldn't rely on human speech to communicate with them. Instead, we created new languages that computers could understand. Arguably, the most important of these languages is HTML. HTML is the foundation of most web pages — it's how we tell browsers to structure content into titles, headings, paragraphs, images, links, lists, forms, tables, and more. In this post, we'll learn what HTML is and what it's used for. Then, we'll walk through how to write your own HTML file and review some of the most common elements and attributes in the language. Finally, we'll end with a brief look at some resources you can use to continue learning and writing HTML. Let's get started. First published by Tim Berners-Lee in 1989, HTML is now used by 92% of all websites, and probably all the ones you visit. HTML is short for "Hypertext Markup Language." Let's break this down word-by-word to better understand what HTML actually means. "Hypertext" is text that contains references to other text or pages, also known as hyperlinks. Hyperlinks allow you to go anywhere on the web with a click of the mouse. Rather than reading a web page in the linear order that the author laid out, like in print, we can use hyperlinks to jump to another section of the same page, a different page on the current website, or to a completely new website. For example, here's a hyperlink that sends readers back to the top of this blog post. Hyperlinks can also open a PDF, email, or multimedia, like a video or audio file. Linking information together in this way revolutionized the web. Together, HTML and the internet make it possible for anyone to access all types of information around the world, in any order they want. "Markup" refers to how HTML "marks up" the page with annotations within the HTML file. These annotations are not displayed on the web page itself — they work behind-the scenes telling browsers how to display the document to visitors. We'll learn more about this markup soon. "Language" is the simplest part of the acronym to understand. Like any language, HTML is made up of a unique syntax and alphabet. But what kind of language is it, exactly? Let's tackle this question below. Is HTML a programming language? Whether HTML is or is not technically a programming language is an ongoing debate among web developers and experts. The majority defines HTML as a markup language, not a programming language, although some argue the two aren't mutually exclusive. To understand this distinction, we have to know the definition of a programming language. All programming languages have some functional purpose — they need to "do" something, whether it be evaluating expressions, declaring variables, or modifying data. These languages not only instruct computers what to do, but how to do it. JavaScript is the most widely-used programming language in web development. Other popular programming languages include Python, Java, and C/C++. HTML, on the other hand, doesn't really "do" anything. It simply gives browsers the content it needs to display. HTML doesn't care how the browser goes about displaying the content, as long as it's displayed. In other words, HTML has a structural purpose, not a functional one. Still, some developers use this same logic to argue that HTML is a programming language — it's just a declarative programming language. According to Professor David Brailsford from the University of Nottingham, for example, declarative languages are much more restricted than other languages because they ask for something and don't care how it happens, but that doesn't make them any less of a programming language. While this is an interesting and rich point of discussion, it probably won't affect how you code in HTML. With that in mind, let's move on to what HTML is used for. What is HTML used for? HTML is primarily used for creating web pages. Because it's open-source and supported by all modern browsers, HTML is free to use and ensures your text, images, and other elements are displayed intended. Without HTML, all web pages would be plain text files that looked like this: With HTML, not only can you format documents with headings, paragraphs, lists, and other elements — you can also embed images, videos, audio files, and other multimedia via hyperlinks. And, you can link to other web pages on the same website or from another site. This allows visitors to easily navigate your website and jump between websites stored on different web servers. Even after adding headings, images, and hyperlinks, you'd still have a very basic web page — and that's by design. HTML is supposed to create a simple base upon which Cascading Style Sheets (CSS) and JavaScript (JS) can be added. With CSS, you can customize your styling and layouts, changing the color, font, and alignment of elements. With and JS, you can add dynamic functionality like pop-ups and photo sliders. HTML is used to create things other than web pages, too. You can use it to make tables for organizing data. You can create forms for collecting user information, processing transactions, making reservations, or placing an order. You can also create emails with HTML. Whether you want to create web pages, tables, forms, or emails, you'll need to know how to write HTML. Let's break down the process below. How to Write HTML As mentioned, HTML is just plain text annotated with markup. More precisely, this markup consists of tags and attributes. To help you visualize this syntax, here's a graphic from Online Design Teacher: Image Source Let's going to take a closer look at each component of an element below. HTML Tags HTML elements are designated by tags. Most elements have an opening and closing tag. Opening tags precede the text and contain the element name enclosed by the brackets "". Closing tags are identical to opening tags, save for a backward slash that precedes the element name. Say you want to add a paragraph to your web page, and the text of the paragraph is "This is a paragraph." You'll wrap it with the HTML paragraph tags: . So, the HTML will look like this: This is a paragraph. Together, these three things are all you need make a paragraph element in HTML. Most HTML elements are the same: an opening tag, a closing tag, and content in between. Some HTML elements, such as
(break), only have an opening tag — these are called empty tags. Element names are case-insensitive. Meaning, they can be written in uppercase, lowercase, or some combination of the two. For example, the tag can also be written as . However, it is considered a best practice to always write the name in lowercase. HTML Attributes While all HTML elements need tags, only some need attributes. An attribute provides additional information about the HTML element, and this information can be essential or non-essential. For example, an image element must always have a source attribute whose value is the image URL or file path. Otherwise, the browser will not know what image to render. The same goes for the anchor element, which is used to create hyperlinks — it must contain an href attribute whose value specifies the link's destination. Otherwise, if a visitor clicks on the anchor element, the browser won't send them anywhere. Other attributes aren't essential to include but are considered a good practice. For example, a browser can render an image without the alt attribute, which contains image alt text. But, a reader with visual impairments might have trouble understanding what the image conveys without an alternative text description. Now that we understand the importance of attributes, let's make sure we understand how to find and write them. An attribute is always found in the opening tag of an HTML element and has the syntax: name="value". Many elements have their own set of attributes that affect how the content is rendered on the page. Attributes can be written in any order inside the opening tag. However, you cannot put multiple instances of the same attribute inside the same HTML tag. How to Create an HTML File To build a website with HTML, you need to create an HTML file first. This file will contain all the HTML for your web page, and will be uploaded to your web server. That way, when a visitor searches for your website, the server will send the HTML file to the visitor's browser, and the browser will render the page accordingly. Since an HTML file is in standard text format, you can use basic text editors like Notepad++ to create and edit your file. Once you select an editor, writing the actual code is straightforward. We'll walk through the process step-by-step so you can create an HTML file for your web project. To start, you need declare the type of document as HTML. To do so, add the special code on the very first line of the file. Next, you'll want to define the root element of the document. Since this element signals what language you're going to write in, it's always going to be in an HTML5 doc. On the next line, add an opening tag. Below that, add a closing tag. Within the opening tag of the html element, you should also include a lang (language) attribute. This will help screen readers determine what language the document is in, making your website more accessible. Without a language attribute, screen readers will default to the operating system's language, which could result in mispronunciations of the title and other content on the page. Since we're writing this post in English, we'll set the file's lang attribute value to "en." An HTML doc is composed of two parts: the head section and the body section. The head contains meta-information about the page as well as any internal CSS. The browser does not show this information to readers. The body section contains all the information that will be visible on the front end, like your paragraphs, images, and links. To create these sections, add a tag and then a tag between and in your document. In the head section, you'll want to name your document. Write a name (we'll go with "My HTML Page" in this example) and then wrap it in tags. I'm also going to add tags inside the head section. Between these tags, you'd add any internal CSS you're using to style your HTML. Instead of making up some CSS rules, I'm just going to add a comment in CSS as a placeholder. You can add commends in HTML too for anything you don't want the browser to render. Note: In many HTML documents, you won't see style tags or any CSS. That most likely means the page is using an external style sheet, a common way of adding CSS to HTML. In the case of an external stylesheet, you'll see a link in the head section of the doc. In the body section, let's now add a heading and paragraph. You'll write out the heading name and wrap it in tags, and write out the paragraph and wrap it in tags. Finally, it's a common practice to indent nested HTML elements. Even though this makes not difference to the browser processing the file, indenting visualizes the document's structure and makes it easier to read. HTML Example All together, this is what your HTML file will look like: My HTML Page /* These style tags are only necessary if you're adding internal CSS */ This is a Heading This is a paragraph. Below is how it would look on the front end. Note that only the heading and paragraph from the body section are rendered. See the Pen Simple HTML Page by Christina Perricone (@hubspot) on CodePen. As you can see, this is a pretty skeletal HTML file. To fill it in, we need to take a look at some more common HTML elements. We've already noted two, the and elements. Let's take a closer look at these elements and others below. Common HTML Elements The first version of HTML consisted of just 18 tags. Since then, four versions have been released with dozens of tags added in each version. In the most recent version, HTML5, there are 110 HTML tags. Below we'll review the most common elements and their tags. Paragraph () The HTML paragraph element represents a paragraph. By placing tags around text, you'll make that text start on a new line. Here's an example of two paragraphs: See the Pen p example by Christina Perricone (@hubspot) on CodePen. Image () The HTML image element embeds an image into the document. It requires a src (source) attribute to render properly. An alt attribute should also be included in case the image doesn't load properly or the reader has a visual impairment. Here's an example of an image with a source and alt attribute: See the Pen img example by Christina Perricone (@hubspot) on CodePen. Headings (-) The HTML heading elements represent different levels of section headings. is the highest section level and the most prominent, whereas is the lowest and therefore least prominent. See the Pen headings example by Christina Perricone (@hubspot) on CodePen. Division () The HTML content division (div) element is a generic block-level container for "flow content." Flow content is a category of HTML elements that contain text or embedded content. The anchor, block quote, and heading elements are considered flow content. On the back end, div elements help organize the code into clearly marked sections. On the front end, they add line breaks before and after their content. Otherwise, they do not affect the content or layout of the page unless styled with CSS. Here's an example of div wrapped around an image: See the Pen div example 1 by Christina Perricone (@hubspot) on CodePen. Here, the image looks the same as it did without the div wrapper element. That's because no style information was given to this div element. To change the appearance of the container and therefore the image inside that container, you'd need to add style information. Say, for example, you wanted to center the image. Then you could use the following code to horizontally center the image on the page: See the Pen div example 2 by Christina Perricone (@hubspot) on CodePen. Span () The HTML span element is a generic inline container for "phrasing content." Phrasing content refers to text and any markup it contains, like and tags. Span tags do not inherently represent anything, but they are used to group phrasing content for two reasons. The first is to apply style information to a particular piece of text. For example, if you're creating drop caps, you can wrap the first letter of the opening paragraphs of each section of your article in span tags. See the Pen span example by Christina Perricone (@hubspot) on CodePen. The second reason to use span tags is to group elements that already share attribute values. For example, maybe you have a website for English speakers learning French. The default language is set to English but on several pages, you might have a table with French terms in the first column and their English translations in the second column. In that case, you can wrap the French terms in span tags with the language attribute set to "fr." Anchor () The HTML anchor element creates a hyperlink. The anchor element requires an href attribute, which species the destination of the link. The destination can be another section on the same web page or another web page on the same site, or external websites, files, and email addresses. Here's an example of a link nested in a paragraph: See the Pen anchor example by Christina Perricone (@hubspot) on CodePen. Unordered List () The HTML unordered list element is used for grouping items when the order doesn't matter. Shopping lists, for example, don't need to follow a particular order. List items must be defined by the tag and then wrapped in the element. Here's an example of an unordered list. Try adding some list items yourself and see how the list changes. See the Pen ul example by Christina Perricone (@hubspot) on CodePen. Ordered List () The HTML ordered list element is used for grouping items when the order does matter. Recipes, for example, should follow a particular order. The steps must be defined by the tag and then wrapped in the element. An ordered list will start at the number 1 by default. If you'd like to start at another number, simply add a start attribute and set the value to the number you want. Here's an example of an ordered list that starts at 1. Try adding steps at different parts of the list: See the Pen ol example by Christina Perricone (@hubspot) on CodePen. Emphasis () The HTML emphasis element emphasizes the text it contains. Browsers typically render the text in italics. Here's an example of the emphasis wrapped around a paragraph and nested within a paragraph: See the Pen em example by Christina Perricone (@hubspot) on CodePen. Strong () The HTML Strong element indicates that the text it contains is of strong importance or urgency. Browsers typically render the text in bold. Here's an example of the strong element: See the Pen strong example by Christina Perricone (@hubspot) on CodePen. Common HTML Attributes Attributes modify HTML elements in different ways. They can change the appearance of the element, apply unique identifiers so the elements can be targeted by CSS, or provide necessary information to readers or screen readers. Below we'll take a look at the most common attributes. Style Attribute The style attribute contains inline CSS. This CSS will override any styles set in the head section of the document or in an external stylesheet. It will only be applied to the HTML element that has the style attribute in its opening tag. Here's an example of the attribute in HTML: This paragraph will be black by default. This paragraph will be maroon. ID Attribute The ID attribute is used to identify a single element in an HTML file. That means the value of an ID attribute should not be repeated inside the same file. Using this unique value, you can target a single element with internal or external CSS. Here's an example of the attribute in HTML: Title in Fancy Typography Class Attribute The class attribute is used to identify a group of elements under the same name and customize that group, effectively creating a new group of elements. Bootstrap buttons, for example, are all labeled with the .btn class so they have the same basic style: 14px font, medium size, rounded edges, etc. Here's an example of the attribute in HTML: Button Language Attribute As mentioned, the language attribute signals to screen readers what the primary language of the web page is and when they need to switch to another language. This is a small detail that can make your content more accessible to all readers, no matter what region they're from or the language they speak. While this attribute is most commonly embedded in the HTML element, it can also be used with paragraph, div, span, and other elements. Here's an example of the attribute in HTML: Cette phrase est en français Esta frase es en español. Href Attribute An href attribute contains the destination of a link. This attribute must always be included with an anchor element. Here's an example of the href attribute in HTML: go to HubSpot.com Source Attribute Just like an anchor element needs an href attribute, an image needs a source attribute. This contains the path to the image file or its URL. Here are two examples of the attribute in HTML: Data-* Attribute The data-* attribute is used to store custom data that is private to the page or application. You can use this stored data in the document's JavaScript to create a more dynamic experience for the user. The asterisk in the data-* attribute can be any value. Here's an example of the attribute in HTML from W3Schools: Owl Salmon Tarantula I could then use this data in the JavaScript to trigger a pop-up message providing more information about each list item. Say, for example, a visitor clicked on the word "Owl." Then a pop-up box would appear saying "The owl is a bird." Now that we've covered the most common elements and attributes in HTML, let's explore where you can practice writing this language and continue learning more about it. How to Learn HTML There are hundreds of resources available to learn HTML. Depending on your learning style, you may prefer to read blog posts, watch video tutorials, take online courses, download an ebook, or use a combination of all of these resources. Below we'll walk through at least one example of each of these content types. That way, no matter what type of learner you are, you can find a resource that will help you learn this programming language. 1. The Beginner's Guide to HTML and CSS for Marketers This free e-book will explain what HTML and CSS are, how they're different, and how to get started if you're brand new at coding. As the title suggests, this guide is specifically designed for marketers who need to be able to make quick fixes to their websites, blogs, and landing pages. 2. Lynda.com If you're a visual learner, check out the online tutorials available at Lynda.com. Lynda.com offers 48 courses and over 1,000 video tutorials that cover virtually every HTML topic, from forms to semantic data to everything in between. These lessons are organized into three levels — beginner, intermediate, and advanced — so you can develop your skills over time. To get access to all content on the site, you can sign up for a monthly or annual subscription. 3. Codecademy If you're overwhelmed by the sheer quantity of videos available on Lynda.com, try Codeacdemy's Learn HTML class. This online course will start with the basic structure and elements of HTML. You can then put your knowledge to the test by building out more complex elements and projects, including HTML tables and forms, from scratch. While you can complete most of the course for free, there are pro features like quizzes and projects that you'll have to pay to unlock. 4. W3Schools HTML Tutorial Sometimes you have to learn by doing. W3Schools HTML Tutorial is centered on that exact concept. With its online code editor, you can start with the bare bones of an HTML document and practice writing HTML from scratch, or you can start with more fully fleshed out examples and edit them. You can also complete exercises and quizzes for each topic you cover. 5. Learn HTML Like W3Schools, Learn HTML is a free interactive tutorial. However, rather than try to be the most comprehensive resource on HTML, Learn HTML offers a short step-by-step guide for building out a web page. At every step, you can test whether you understood the lesson by completing an exercise in the online code editor. If your code matches the expected output, then you'll get a success message and be invited to move onto the next lesson. Now you're ready to code. HTML is the language we use to talk to computers. It is how browsers display text, images, paragraphs, and other elements on a web page. As such, HTML is the predominant language of the World Wide Web. That makes the language important not only to people trying to become programmers, but to marketers like yourself. Knowing the basics of this markup language will allow you to make changes to your website without needing to rely on a developer, saving you and your business time and money. Editor's note: This post was originally published in August 2020 and has been updated for comprehensiveness. Originally published May 6, 2021 7:00:00 AM, updated May 10 2021