

Lecture 1

HTML



Intro to HTML

Topics

- **Introduction**
- **History of JavaScript**
- **Browser Debugging Tools**
- **The JavaScript Console**
 - **Console.log**
 - **Syntax Examples**

World Wide Web

Invention of WWW



Tim Berners-Lee, a British scientist, invented the World Wide Web (WWW) in 1989, while working at CERN. The Web was originally conceived and developed to meet the demand for automated information-sharing between **scientists** in universities and institutes around the world.

HTML (Hyper Text Markup Language) was first developed by Tim Berners-Lee in 1990. He is the primary author of HTML with the assistance of his colleagues at CERN, an international scientific organization based in Geneva.

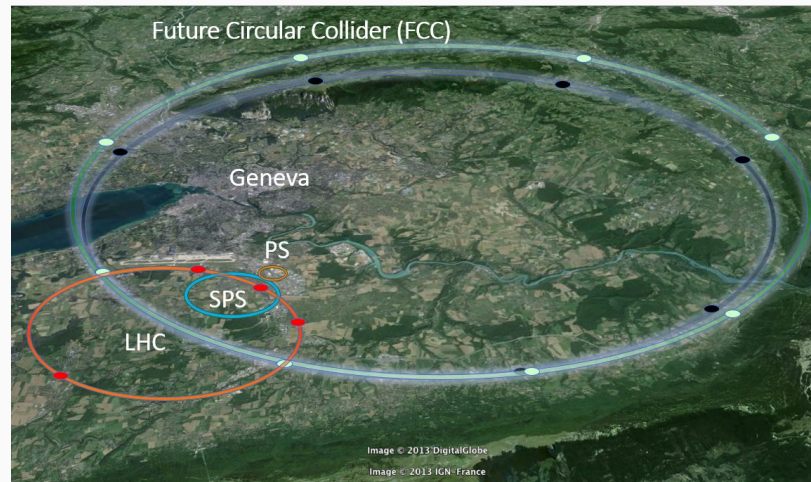
He invented HTML for converting any word document into a file viewable on the Internet, thus it will be available to anybody with an internet connection. It is also used to define the structure and layout of a web page.

What is CERN?



The European Organization for Nuclear Research, known as CERN, is a European research organization that operates the largest particle physics laboratory in the world. Established in 1954, located in Geneva on Franco-Swiss border.

Home of the Atom Smasher (Large Haron Collider)



History of World Wide Web

WWW



1989

Tim Berners-Lee
Invents the World
Wide Web

1990

Tim Berners-Lee
develops HTML
code

1991

The first live
website is
launched!



Google



mid 2000s

Adaptation of
the mobile
phones takes
hold.

1998

Google officially
launches!

1996

Cascading Style
Sheets are
released.

History of World Wide Web



2007

Mobile Safari launches as the first mobile browser.



2010

Ethan Marcotte coins the term "Responsive Web Design"



2015

Google sets mobile friendly deadline.

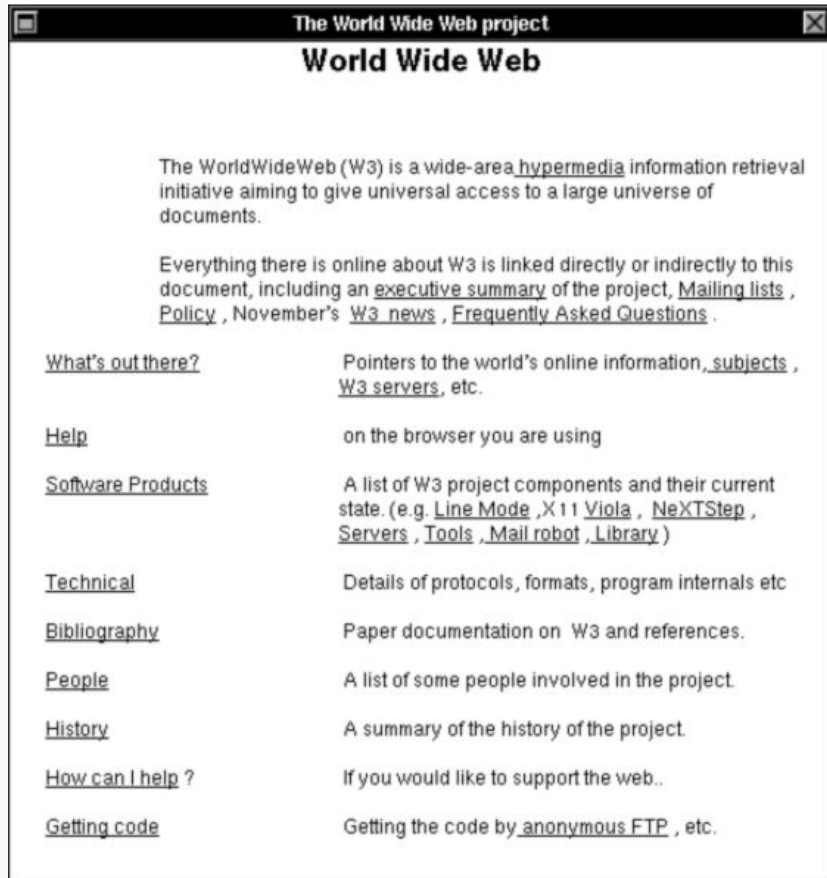
27

Years of existence

HTML History

Year	Version
1989	Tim Berners-Lee invented www
1991	Tim Berners-Lee invented HTML
1993	Dave Raggett drafted HTML+
1995	HTML Working Group defined HTML 2.0
1997	W3C Recommendation: HTML 3.2
1999	W3C Recommendation: HTML 4.01
2000	W3C Recommendation: XHTML 1.0
2008	WHATWG HTML5 First Public Draft
2012	<u>WHATWG HTML5 Living Standard</u>
2014	<u>W3C Recommendation: HTML5</u>
2016	W3C Candidate Recommendation: HTML 5.1
2017	<u>W3C Recommendation: HTML5.1 2nd Edition</u>

First Static Web Sites



The first web sites on the World Wide Web were writing in very basic mark up using HTML 1.

Navigating a web page on the initial version of NXT machine first browser was a very static experience compared to rich user experience we have today.

Navigate the web using the first browser here:

<https://worldwideweb.cern.ch/>

What is HTML?

What is HTML?

HTML (Hypertext Markup Language) is **not** a programming language. It is a ***markup language*** that tells web browsers how to structure the web pages you visit. It can be as complicated or as simple as the web developer wants it to be. HTML consists of a series of **elements**, which you use to enclose, wrap, or *mark up* different parts of content to make it appear or act in a certain way. The enclosing **tags** can make content into a hyperlink to connect to another page, italicize words, and so on. For example, consider the following line of text:

```
1 | My cat is very grumpy
```

If we wanted the text to stand by itself, we could specify that it is a paragraph by enclosing it in a paragraph (**<p>**) element:

```
1 | <p>My cat is very grumpy</p>
```

HTML Tags

Note: Tags in HTML are **case-insensitive**. This means they can be written in uppercase or lowercase.

For example, a `<title>` tag could be written as `<title>`, `<TITLE>`, `<Title>`, `<TiTlE>`, etc., and it will work. However, it is best practice to write all tags in lowercase for consistency, readability, and other reasons.

HTML Recap

HTML



- HTML stands for Hyper Text Markup Language
- HTML is the standard markup language for creating Web pages
- HTML describes the structure of a Web page
- HTML consists of a series of elements
- HTML elements tell the browser how to display the content
- HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.

Web Browsers



- Websites run in browsers, which means you need some browsers to run websites!
- Your users could be using **any** browser to visit the websites that you build for them. There are slight differences in the way that different browsers will render your websites, which means that you'll need to test your sites in *all* of them.
- Now, you don't need to download every browser right now, but I want to make sure that you install the two most popular cross-platform browsers: **Google Chrome** and **Mozilla Firefox**.
- Other browsers, like **Apple Safari** and **Microsoft Edge**, are fantastic as well and you should continue to browse and test your sites with them.

Web Browser cont..

- The purpose of a web browser (Chrome, Edge, Firefox, Safari) is to read HTML documents and display them correctly.
- A browser does not display the HTML tags, but uses them to determine how to display the document:

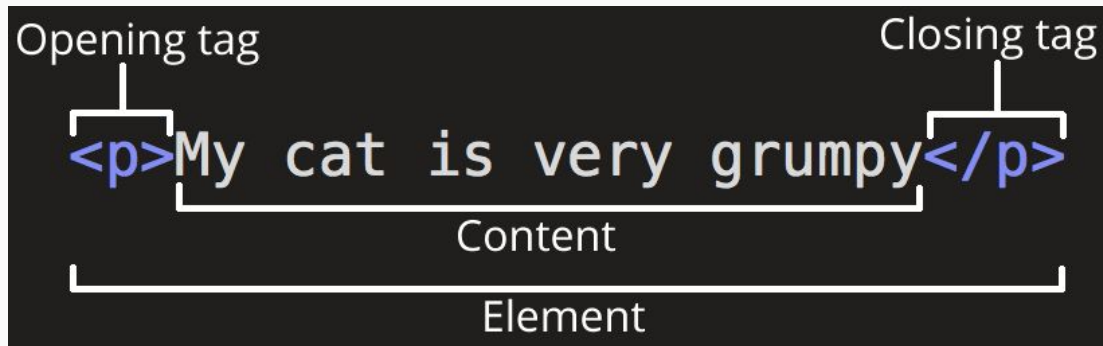


HTML Syntax

Aatomy of HTML Element

- **The opening tag:** This consists of the name of the element (in this example, *p* for paragraph), wrapped in opening and closing angle brackets. This opening tag marks where the element begins or starts to take effect. In this example, it precedes the start of the paragraph text.
- **The content:** This is the content of the element. In this example, it is the paragraph text.
- **The closing tag:** This is the same as the opening tag, except that it includes a forward slash before the element name. This marks where the element ends. Failing to include a closing tag is a common beginner error that can produce peculiar results.

The element is the opening tag, followed by content, followed by the closing tag.



HTML Element Examples

Some HTML elements have no content (like the `
` element).

These elements are called empty elements. Empty elements do not have an end tag!:

Start tag	Element content	End tag
<code><h1></code>	My First Heading	<code></h1></code>
<code><p></code>	My first paragraph.	<code></p></code>
<code>
</code>	<i>none</i>	<i>none</i>

Nesting Elements

Elements can be placed within other elements. This is called *nesting*. If we wanted to state that our cat is **very** grumpy, we could wrap the word *very* in a `` element, which means that the word is to have strong(er) text formatting:

```
<p>My cat is <strong>very</strong> grumpy.</p>
```

There is a right and wrong way to do nesting. In the example above, we opened the `p` element first, then opened the `strong` element. For proper nesting, we should close the `strong` element first, before closing the `p`.

The following is an example of the *wrong* way to do nesting:

```
<p>My cat is <strong>very grumpy.</p></strong>
```

The **tags have to open and close in a way that they are inside or outside one another**. With the kind of overlap in the example above, the browser has to guess at your intent. This kind of guessing can result in unexpected results

Block vs Inline Elements

There are two important categories of elements to know in HTML: block-level elements and inline elements.

- **Block-level elements** form a visible block on a page. A block-level element appears on a **new line** following the content that precedes it. Any content that follows a block-level element also appears on a new line. Block-level elements are usually structural elements on the page. For example, a block-level element might represent headings, paragraphs, lists, navigation menus, or footers. A block-level element wouldn't be nested inside an inline element, but it might be nested inside another block-level element.
- **Inline elements** are contained within block-level elements, and surround only small parts of the document's content. (not entire paragraphs or groupings of content) An inline element will not cause a new line to appear in the document. It is typically used with text. For example, as an `<a>` element (hyperlink) or emphasis elements such as `` or ``.

Block vs Inline Elements cont..

```
1 | <em>first</em><em>second</em><em>third</em>
2 |
3 | <p>fourth</p><p>fifth</p><p>sixth</p>
```

- `` is an inline element. As you see below, the first three elements sit on the same line, with no space in between. On the other hand, `<p>` is a block-level element. Each `p` element appears on a new line, with space above and below. (The spacing is due to default [CSS styling](#) that the browser applies to paragraphs.)

firstsecondthird

fourth

fifth

sixth

HTML Comments

- HTML has a mechanism to write comments in the code. Browsers ignore comments, effectively making comments invisible to the user. The purpose of comments is to allow you to include notes in the code to explain your logic or coding. This is very useful if you return to a code base after being away for long enough that you don't completely remember it. Likewise, comments are invaluable as different people are making changes and updates.
- To write an HTML comment, wrap it in the special markers `<!--` and `-->`. For example:

```
1 | <p>I'm not inside a comment</p>
2 |
3 | <!-- <p>I am!</p> -->
```

I'm not inside a comment

Empty Elements

- Not all elements follow the pattern of an opening tag, content, and a closing tag. Some elements consist of a single tag, which is typically used to insert/embed something in the document. For example, the `` element embeds an image file onto a page:

```
1 | 
```

The following output would be



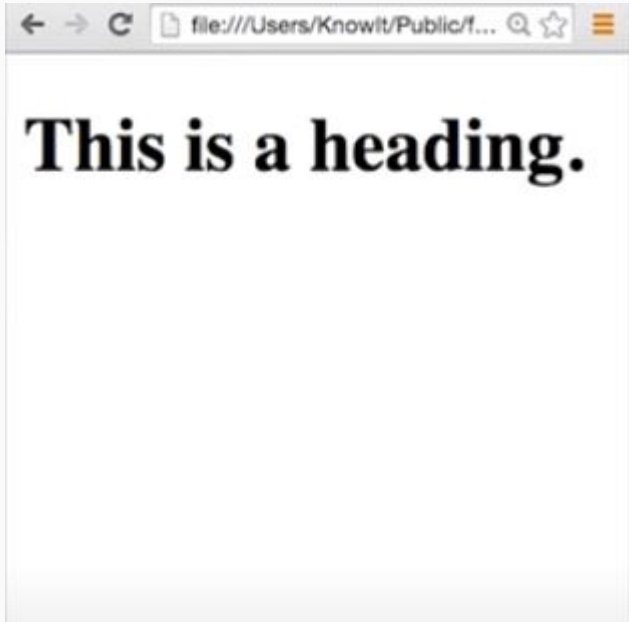
HTML Structure

What is HTML Structure?

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>HTML Structure</title>
  </head>
  <body>
    <h1>This is a heading.</h1>
  </body>
</html>
```

- The `<!DOCTYPE html>` declaration defines that this document is an HTML5 document
- The `<html>` element is the root element of an HTML page
- The `<head>` element contains meta information about the HTML page
- The `<title>` element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)
- The `<body>` element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.
- The `<h1>` element defines a large heading

HTML Structure Example



```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>HTML Structure</title>
  </head>
  <body>
    <h1>This is a heading.</h1>
  </body>
</html>
```

HTML Page Structure

```
<html>
```

```
<head>
```

```
<title>Page title</title>
```

```
</head>
```

```
<body>
```

```
<h1>This is a heading</h1>
```

```
<p>This is a paragraph.</p>
```

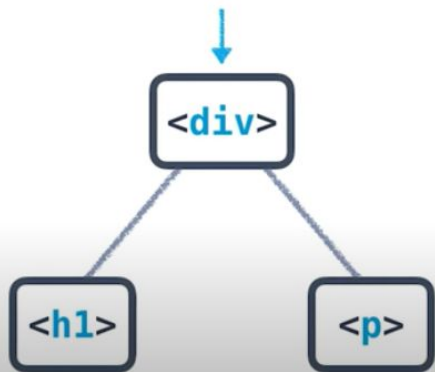
```
<p>This is another paragraph.</p>
```

```
</body>
```

```
</html>
```

HTML and Trees

```
<div>content</div>
```



- Elements can go inside other elements.

```
<div><h1>Article Title</h1></div>
```

- Content isn't just text, it is also other HTML elements
- Division elements (`<div></div>`) are used to group chunks of content together
- In the example, `<h1>` is the child element of the `<div>` element
- We can reorganize this, so it is the structure is more clear

```
<div>  
  <h1>Article Title</h1>  
</div>
```

HTML Document

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>HTML Structure</title>
  </head>
  <body>
    <h1>This is a heading.</h1>
  </body>
</html>
```

An HTML Document can be thought of as a Template.

This template can be broken down into 3 parts:

1. **DOCTYPE**: Describes the type of HTML. While there are technically different types, for 99.999% of the HTML you'll write, you'll likely be fine with `<!DOCTYPE html>`.
2. **<head>**: Describes meta information about the site, such as the title, and provides links to scripts and stylesheets the site needs to render and behave correctly.
3. **<body>**: Describes the actual content of the site that users will see.

HTML Document

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>HTML Structure</title>
  </head>
  <body>
    <h1>This is a heading.</h1>
  </body>
</html>
```

Omitting some of this information doesn't necessarily mean that the page won't be displayed. In fact, your browser will assume certain parts of the template exist even if you accidentally leave them out.

Take this line of HTML for example:

```
<h1>This is a heading</h1>
```

If you create an HTML file with only this line, open the file in any modern browser, and inspect the page with developer tools, you'll see that certain parts of the basic HTML document format were assumed:

```
<html>
  <head></head>
  ▼ <body>
...   <h1>This is a test</h1> == $0
      </body>
</html>
```

Developer Tools - https://developers.google.com/web/tools/chrome-devtools/

Elements Console Sources Network Performance Memory Application Security »

```
<!DOCTYPE html>
<html lang="en" class="chekov">
  <head>_</head>
  <body class="devsite-doc-page">
    <div class="devsite-header-no-lower-tabs no-touch" id="top_of_page"> == $0
    <div class="devsite-wrapper" style="margin-top: 48px;">_</div>
      <span id="devsite-request-elapsed" data-request-elapsed="214.869976044"></span>
      <iframe src="https://clients5.google.com/pagead/drt/dn/" aria-hidden="true" style="display: none;">_</iframe>
```

html.chekov body#top_of_page.devsite-doc-page.devsite-header-no-lower-tabs.no-touch

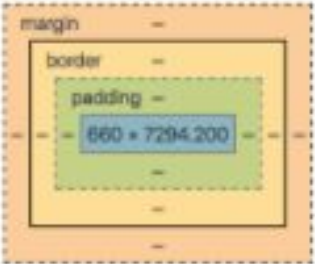
Styles Event Listeners DOM Breakpoints Properties

Filter :hov .cls +

```
element.style {
}

body, html {
  color: #212121;
  font: 400 16px/24px Roboto,sans-serif;
  -moz-osx-font-smoothing: grayscale;
  -webkit-font-smoothing: antialiased;
  margin: 0;
  -webkit-text-size-adjust: 100%;
  -ms-text-size-adjust: 100%;
  text-size-adjust: 100%;
}
```

body div dl dd devsite-google-blue.css:1



margin border padding 660 x 7294.200

Filter ☐ Show all

HTML Doctypes

HTML Doctypes

An HTML document will usually start with a type declaration (which is not a tag, so it should not have a closing tag). The declaration helps the browser determine what type of HTML document it's trying to parse and display.

If you've ever looked at an [older website](#) using dev tools, you might have noticed a doctype that looks like this:

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

Or maybe you don't see a doctype at all.

```
<html>
```

```
...
```

```
</html>
```

(Triggers “Quirks” mode. This is bad.)

HTML Doctypes cont..

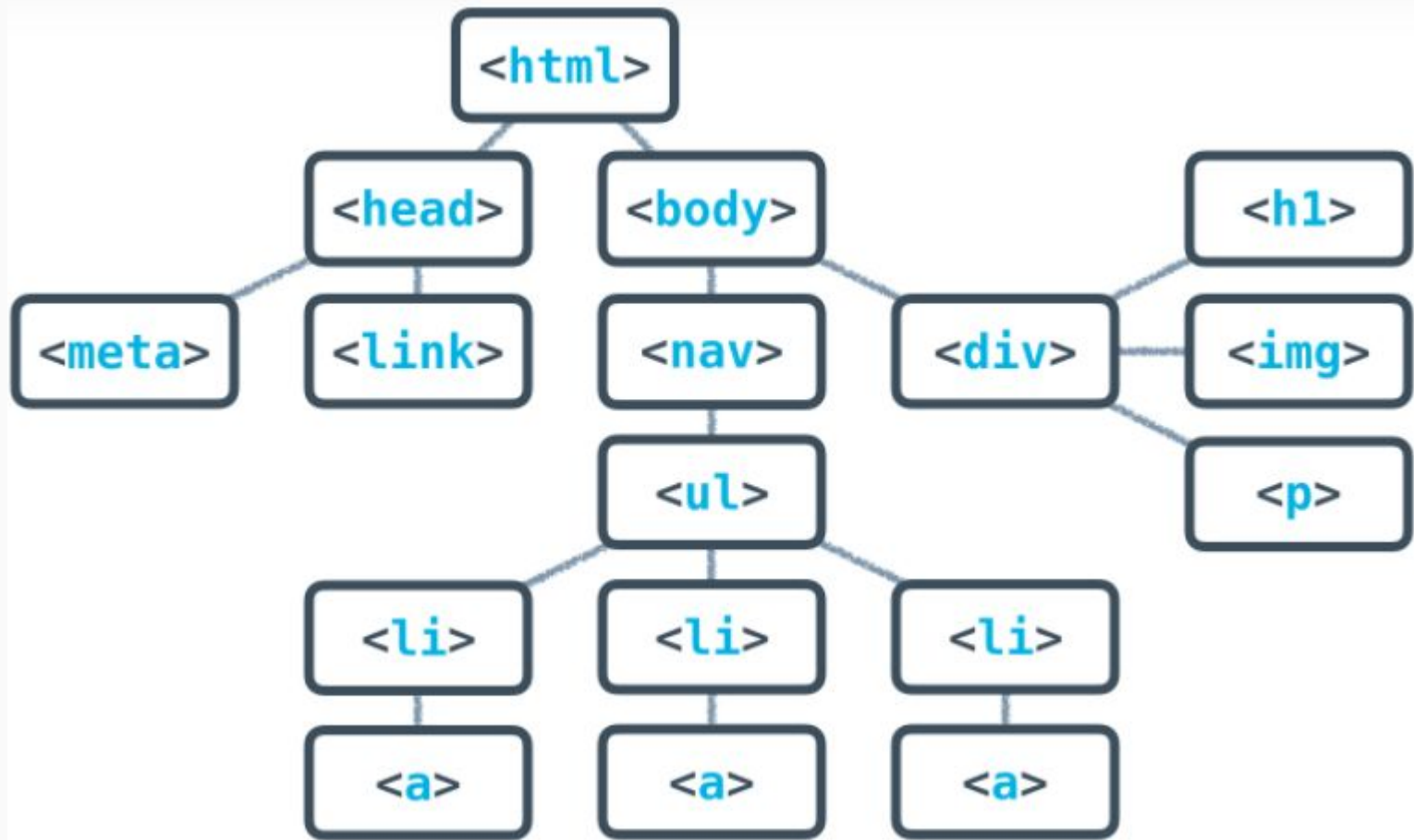
But newer websites (and your websites!) will have a declaration that looks like this:

```
<!DOCTYPE html>
```

(Triggers Standards mode with all updated features.) 😊

Browsers look for this doctype declaration to determine which **rendering mode** to use to render the site. Generally, newer sites follow standard HTML specifications. The current standard HTML specification is called HTML5 (which is what you're learning!). On the other hand, older sites, created before HTML standards really existed, might use a different rendering mode that imitates the behavior of older browsers.

Once you've declared the doctype, the next part of your HTML document is the `<html>` tag, which tells the browser that everything enclosed inside the `<html> ... </html>` should be parsed as HTML. Then you have the two main sections of your HTML document: `<head>` and `<body>`



HTML Documents In Depth

All of the HTML syntax that you've learned in this lesson will help you create the **content** of the page, which is always contained inside the `<body>` tags. The `<body>` is always visible.

The `<head>`, on the other hand, is never visible, but the information in it describes the page and links to other files the browser needs to render the website correctly. For instance, the `<head>` is responsible for:

- the document's title (the text that shows up in browser tabs): `<title>About Me</title>`.
- associated CSS files (for style): `<link rel="stylesheet" type="text/css" href="style.css">`.
- associated JavaScript files (multipurpose scripts to change rendering and behavior): `<script src="animations.js"></script>`.
- the charset being used (the text's [encoding](#)): `<meta charset="utf-8">`.
- keywords, authors, and descriptions (often useful for [SEO](#)): `<meta name="description" content="This is what my website is all about!">`.


Header Tags : Meta and Title

`<meta charset="utf-8">` is pretty standard, and will allow your website to display any [Unicode character](#). ([Read more on how UTF-8 works here.](#)) `<title>` will define the title of the document and will be displayed in the tab of the browser window when a user visits the page.

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>This is a title</title>
  </head>
  <body>
    <h1>Hello!</h1>
  </body>
</html>
```

HTML Validators

This might seem like a lot to remember, but thankfully, there are tools out there to help you. Much like how the Feedback Extension tells you when you've met all the requirements for a particular project, [HTML validators](#) analyze your website and verify that you're writing valid HTML.

**Markup Validation Service**
Check the markup (HTML, XHTML, ...) of Web documents

Validate by URI

Validate by File Upload

Validate by Direct Input

Validate by direct input

Enter the Markup to validate:

[► More Options](#)

Check

QUIZ

1. What is HTML?

- a. An programming language
- b. A Markup language
- c. A database script

1. What is HTML?

- a. An programming language
- b. A Markup language
- c. A database script

Question 2

Will the browser interpret `<HEAD>` differently than `<head>` tag?

- a. No, HTML is case-insensitive
- b. Yes, HTML is not case-sensitive

Question 2

Will the browser interpret `<HEAD>` differently than `<head>` tag?

- a. No, HTML is case-insensitive
- b. Yes, HTML is not case-sensitive

3. What is the anatomy HTML element consist of?

- a. header tag, content tag, footer tag
- b. doctype and meta data
- c. open tag, content, closing tag

Question 3

3. What is the anatomy HTML element consist of?

a. header tag, content tag, footer tag

b. doctype and meta data

c. open tag, content, closing tag

How do we construct comments in HTML?

- a. `/* This is a comment */`
- b. `/// This is a comment`
- c. `<!-- This is a comment -->`

How do we construct comments in HTML?

- a. `/* This is a comment */`
- b. `/// This is a comment`
- c. `<!-- This is a comment -->`

5. How can a HTML Page Structure be visualized?

- a. A Linked Chain structure
- b. A Flat structure
- c. A Tree structure

5. How can a HTML Page Structure be visualized?

a. A Linked Chain structure

b. A Flat structure

c. A Tree structure

Why do we need a doctype declaration?

- a. Browsers need it for the window name and title of window tab.
- b. Browsers need it for unloading of web assets
- c. Browsers need it to determine which rendering mode to render the site with.

Question 6

Why do we need a doctype declaration?

- a. Browsers need it for the window name and title of window tab.
- b. Browsers need it for unloading of web assets
- c. Browsers need it to determine which rendering mode to render the site with.