**BASIC HTML DOCUMENT STRUCTURE**

[SOUND] In this lecture we're going to create a basic HTML document

structure by going straight to the editor and coding one up ourselves and

then we'll validate it on the w3c web validator site.

Okay so I'm in Sublime Text and I have a document open a file

open called .-structure.before.HTML.

And it's located in the examples, Lecture04 folder.

So we have a blank slate in front of us and

we're going to create our very first HTML page.

So every HTML page should start with the doc type or document type declaration.

The words doc type or HTML could be lower or upper case.

The only thing you have to watch out for

is that there shouldn't be any space between less

than exclamation point and the word doc type.

You can have as much space as you want anywhere else but

it just doesn't look that great.

So we'll just keep it to one space.

In the past, these declaration were pretty complicated looking, certainly not too

many people would be able to type them up without copy and paste.

HTML 5 however changed all that.

Now the declaration is as simple as it can be.

All it does is tell the browser that it should get ready to render HTML.

Now I know what some of you might be thinking.

What else would there be if not HTML?

There doesn't seem to be any practical purpose for this declaration.

If you're thinking that, you're absolute right.

This declaration is really largely historical.

When HTML standards were first becoming popular,

the web was full of pages that were not compliant with the standards.

To help browsers render those pages correctly, browsers used the doctype

declaration to distinguish between noncompliant and compliant pages.

Noncompliant pages were rendered in what's called the quirks mode,

and the compliant pages were rendered in what's called the standards mode.

Now, that's all historical.

But what you need to know today is that if you leave off the HTML page declaration,

that will signal to the browser that it should treat your pages

as one not following HTML standard.

I'm not going to go into into what that would actually mean in practice but

needless to say things would be a bit messed up.

Your layout wouldn't work quite right.

The styles you apply would work a bit well quirky.

So to make a long story short always use the simple HTML5 doctype declaration.

Play video starting at :2:34 and follow transcript2:34

Next, goes the html tag,

and that's basically a tag that contains the entire html document.

After the html tag, goes the head tag.

The head tag contains items that describe the main content of the page.

Things like what character coding should the browser use for the main content.

It can contain authors description of the page, page title, and whatever other

external resources are needed to render the page properly, among other things.

The point is it contains some metadata about the main content.

Let's write our first metatag to specify the character set in coding

of our webpage.

While not absolutely required, it's always a good idea to specify the character set

that the browser should know how to interpret the content of the webpage.

The most commonly used character set is UTF 8.

Also note that the meta tag is a stand alone tag.

There is no closing meta tag.

Next we'll specify the title of the page.

The title is one of the tags that is actually required to be here.

Without it, the HTML will be invalid.

Play video starting at :3:40 and follow transcript3:40

After the head tag goes the body tag.

The body tag is the root of all content that is visible to the user.

It is often referred to as a viewport.

We can now write our content.

Play video starting at :3:55 and follow transcript3:55

Okay, so let's take a look at how this looks in the browser.

Play video starting at :4: and follow transcript4:00

Okay, and here's our page.

Let's see the content.

Coursera is so cool.

I'm learning so much.

And you could see that the title, Coursera is Cool, is also there.

Next, let's try to take the code from our page, and copy and

paste it to validate it inside the W3C validator.

Play video starting at :4:21 and follow transcript4:21

And it says we're valid.

Play video starting at :4:25 and follow transcript4:25

I'm sure you've noticed by now that what we're doing is nesting one HTML tag into

another.

So for example, we could say that the head tag contains the title tag.

Play video starting at :4:37 and follow transcript4:37

One important rule when nesting html tags is that you have to

close the last opened tag before you close its parent tag.

If you don't, the html you wrote is invalid.

So for example here I have a paragraph.

And don't worry about what these tags mean at the moment, but the paragraph tag, p,

is closed before the last open tag, span.

Span is close after.

So if we copy and paste this code and place it inside our validator and

check it, you will see that it's complaining, and

saying that, the end tag p seen, but there were open elements.

In other words, it wants us to close the span tag, which was opened last,

before we close the outer, p paragraph tag.

Play video starting at :5:27 and follow transcript5:27

One more note before we move on.

When the browser opens an HTML page.

It always renders or interprets the HTML code sequentially from top to bottom.

So the doctype declaration gets interpreted first, then the HTML tag,

then the head tag, and on and on until it hits the last closing HTML tag.

This'll be important to remember as we progress further into the course.

Play video starting at :5:52 and follow transcript5:52

In summary, we went over the bare minimum HTML document plus weighted sum.

Play video starting at :5:58 and follow transcript5:58

We went over the HTML version declaration.

And remember, we always have to use it,

otherwise the browser will be put into quirks mode and some things will break.

Our first HTML tags.

We used html, meta, head, title, body, and

actually throw in some paragraph tags as well.

We went over the fact that HTML is rendered by the browser sequentially,

meaning top to bottom.

And it renders it top to bottom as well.

Next, we're going to talk about HTML content models.