

IIT Madras
ONLINE DEGREE

Modern Application Development - I
Professor Nitin Chandrachoodan
Department of Electric Engineering
Indian Institute of Technology, Madras
Types of CSS styling and Responsive Websites

Hello, everyone, and welcome to this course on Modern Application Development.

(Refer Slide Time: 00:16)

Inline CSS

- Directly add style to the tag
- Example:

```
<h1 style="color:blue;text-align:center;">A heading</h1>
```

The video shows a man in a checkered shirt speaking, with a browser window in the background displaying the rendered HTML code and the 'Styles' panel showing the inline styles applied to the h1 element.

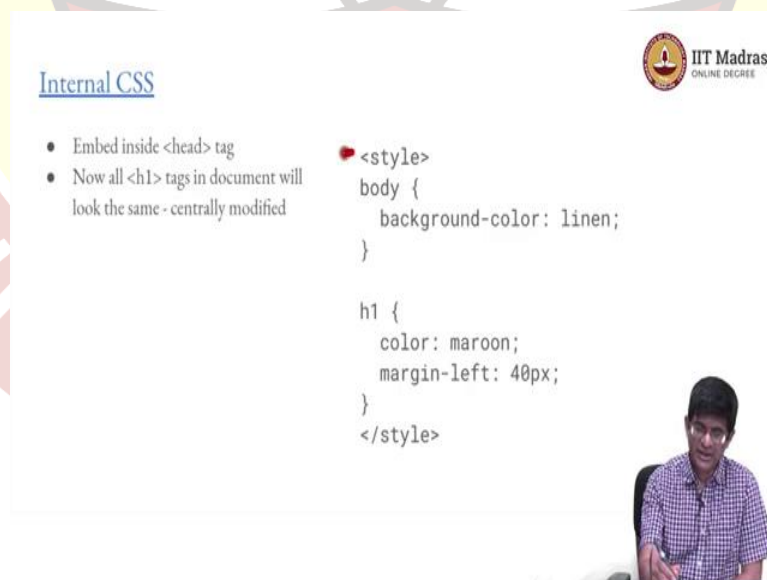
How do I specify styles, if I wanted to override that? One way by which I could do it would directly be to add style to the tag. So, for example, I could have something which directly says,

in the <h1> definition, I put in what are called attributes. So, everything which comes after this h1 over here are attributes that are interpreted as being part of the <h1> definition. And in this case, it is a style attribute which specifies the color to use and how to align it.

And if I do it with this, what I will see is that the output now looks like this. It has now got centered and the color has changed to blue. And in fact, if I go down and look at the styles on the <h1>, I will find that yes, the color:blue and text-align:center both of these were styles that were given directly to the element. This what we have over here is the sort of default <h1> style. This was something that I specified explicitly. And it got applied on top of whatever else was there.

So, you will see that things like the font-size which I did not specify has still been retained, the fact that I want to have a larger font for the heading, the fact that it should be bold has also been retained. Whatever I had over here is now applied on top of that, which means that the color changed and the location. The centering has been changed. That is where the cascading style sheets comes into the picture. I can have one base style sheet. I can layer another one on top of that. And anything which is missing on the top one, it will go and pick it up from one of the underlying layers.

(Refer Slide Time: 01:59)



Internal CSS

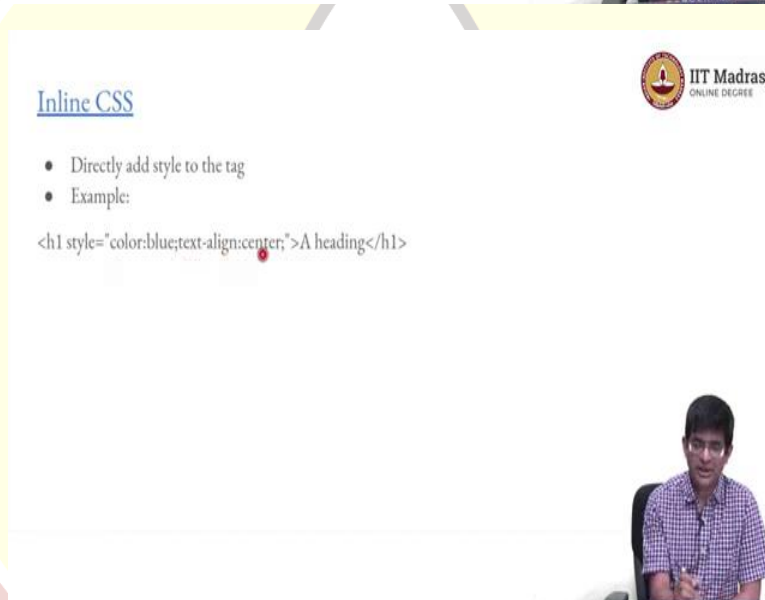
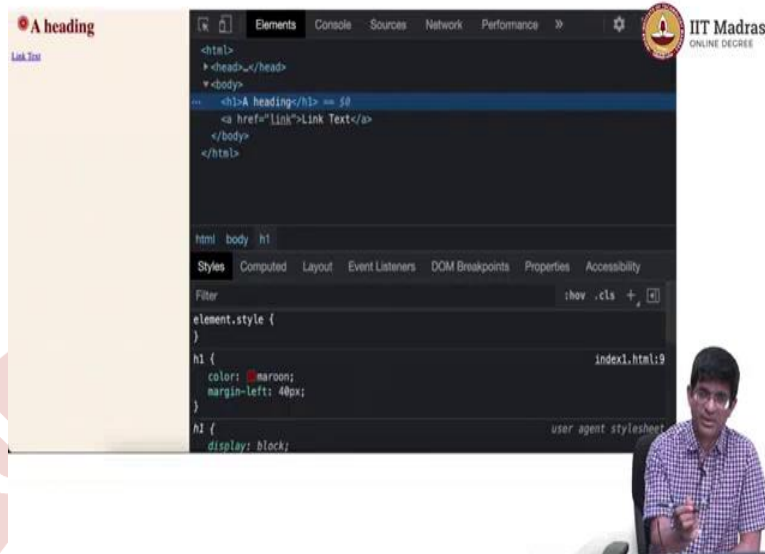
- Embed inside <head> tag
- Now all <h1> tags in document will look the same - centrally modified

```
<style>
body {
  background-color: linen;
}

h1 {
  color: maroon;
  margin-left: 40px;
}
</style>
```

IIT Madras
ONLINE DEGREE

A small video inset in the bottom right corner shows a man with glasses and a checkered shirt speaking.



I could also do this with some kind of internal CSS, where essentially I would say that I have an explicit HTML tag just like head, body, h1 and so on, which defines a region which contains CSS styles. So, what are the styles? The styles basically say what the body color should look like, the body background color, and it also shows how the h1, the first level heading should be displayed. Now, you will notice that inside this <style>, </style>, this part is HTML. Everything else inside it is not really HTML.

It does not follow these angle brackets to define things. It uses its own thing with these curly braces. It also has like some other way by which the body is defined in a certain way, h1 is defined in a certain way. It still seems to have some concept of tags and attributes and so on.

What finally comes out as a result of this will be something like this. You can see over here that I specified the background color should be linen color and the h1 color should be maroon. Yes, that has happened over here. Similarly, the margin on the left hand side should be 40 pixels. It is not centered. So, yes, it has left about 40 pixels on the left hand side before (3:17). And you can see all of those things also showing up down here.

So, in other words, by having this thing inside the `<head>`, so where would this `<style>`, `</style>`, go, it would go inside the `<head>` tag. Now, one good thing about having this kind of styling is, it means that all `<h1>` tags that are present in the document are now going to have their style changed at one shot. In the previous approach, I would have to go and put these attributes into each and every `<h1>` tag. If I make a change somewhere, I have to go and manually change it everywhere. By doing this, I have changed all of them at one shot.

(Refer Slide Time: 04:07)

External CSS

- Extract common content for reuse
- Multiple CSS files can be included
- Latest definition of style takes precedence

IIT Madras
ONLINE DEGREE

सिद्धिर्भवति कर्मजा

Internal CSS

- Embed inside <head> tag
- Now all <h1> tags in document will look the same - centrally modified

```
<style>
body {
  background-color: linen;
}

h1 {
  color: maroon;
  margin-left: 40px;
}
</style>
```



Now, the obvious next step after this is to say, why have this <style>, </style>, inside the HTML document? Why not make it an external file? And the reason behind that is by having it as one external file, I have now opened up the possibility of reuse. I now have a CSS file, which is dedicated just to how pages should look and they can be loaded into several different pages and displayed, so all of those pages get displayed the same way.

This also sort of helps with regard to things like caching of files. The first time I hit that CSS file, I load it, but then for the next page, I do not need to load it again. I know that I already have it with me.

(Refer Slide Time: 04:47)

Responsive Design

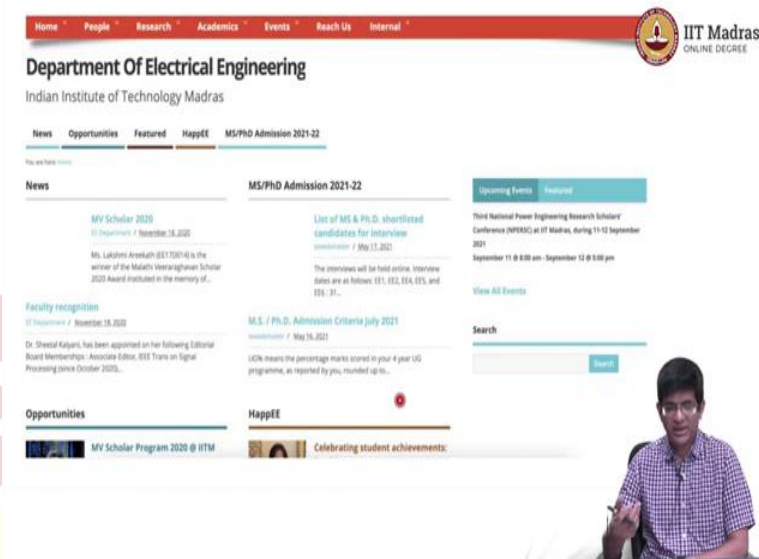
- Mobile and Tablets have smaller screens
 - Different form factors
- Adapt to screen - *Respond*
- CSS control styling - HTML controls content!

IIT Madras
ONLINE DEGREE

So, now with all of this, there is one further aspect of styling that is increasingly important, well, it is very important nowadays and it is not, it is only going to stay as important or become more important as time goes on, which is the fact that a large number of users are actually viewing web pages on mobiles and tablets that have smaller screens than the desktops that we are used to. In general, there are many different form factors that are in use today.

The question then comes, how does a page display across different form factors and will it respond to changes? In other words, let us say I take a phone and I tilt it around into landscape mode, does the page automatically restructure itself, does it try to show me more information or does it just take exactly the same information I had, but just rotated by 90 degrees so that is now visible. And a lot of this can be controlled through CSS styling. The HTML remains completely unchanged. The CSS styling, on the other hand, can decide how the various parts of the page get displayed and shown to you.

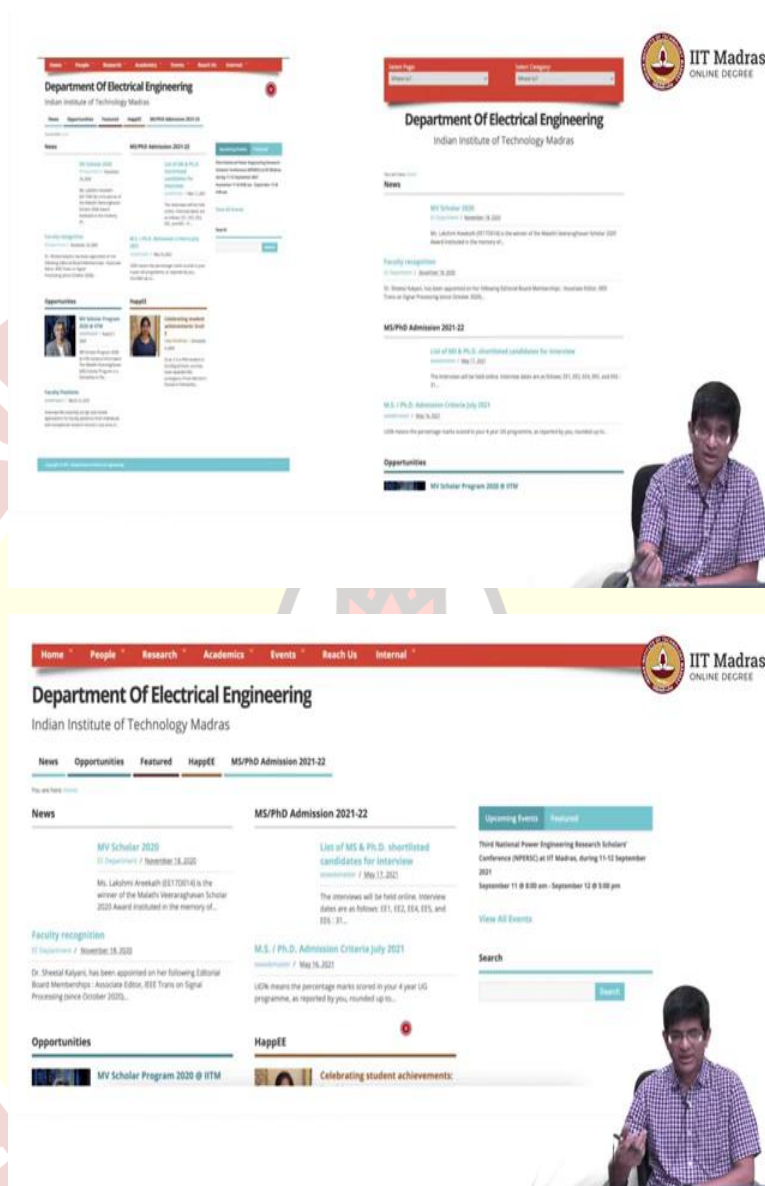
(Refer Slide Time: 06:06)



Now, this is an example. I mean, the, I have just taken a snapshot of the web page of the Department of Electrical Engineering at IIT, Madras. This was taken on a desktop computer. And as you can see, this, it is fairly informative webpage and nice, simple colors. Not too many dramatic colors all over the place. It just has some simple colors. This part over here very clearly shows that it is the heading. You have some part which is just the introduction of the department. And then you have a new section out here and something about events.

So, there is some nice structure to this page. It has things that are laid out appropriately. There are also some navigation tabs over here, news, opportunities, and so on. In other words, structurally, this makes a lot of sense. You look at this page, you at one glance, get a picture of where you might want to go and look for further information. So far, so good. The point is this is on a fairly large screen.

(Refer Slide Time: 07:11)



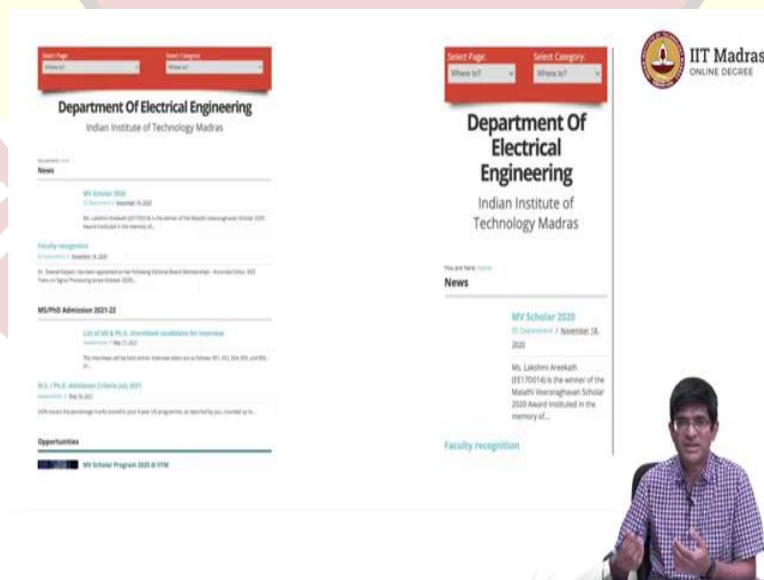
What happens if I was watching it, rather seeing this page on a smaller screen, let us say an iPad in portrait mode. Now, it starts to look a bit more tricky, because now I find that, now how did I get all of these pictures, I basically took a standard browser, went into developer mode and asked him to show me what this would look like on an iPad. And this is what it shows me. It is still not too bad. The problem is that clearly the iPad has a smaller screen. And what it has tried to do is just basically adjust a lot of what it has on to that smaller screen. It has not really fundamentally changed the way the page is displayed in any way.

So, you still have this block on the right over here, which means that there is like these huge chunks of whitespace that are not being used. That was fine on the original desktop, because, well, the desktop has enough space, the iPad does not. Maybe I could have used this in order to make the font a little bit bigger. And other layout, which is by sort of changing it slightly for the iPad is, which happens automatically, I did not really change anything, I just sort of changed what kind of browser I am looking at.

Now, notice that this entire panel that I had on top has got converted into a different sort of header, where I can basically choose which page I want to go to or which category I want to go to. The same way, the news, which was sort of side by side over here has now been replaced one after the other. So, I have these news items one after the other. The upcoming events are there somewhere else, but probably much lower down in the page, maybe that could have been rearranged so that it comes on top. But the point is that this is a lot more readable.

Basically, I can already see this, the font sizes that I have over here and it looks a lot better. So, you can see that, without changing anything, I essentially was able to switch from here to here, basically, because my screen size changed, something triggered a change in which CSS was being used in order to display it and this one actually looks a lot smaller on a constrained screen.

(Refer Slide Time: 09:20)



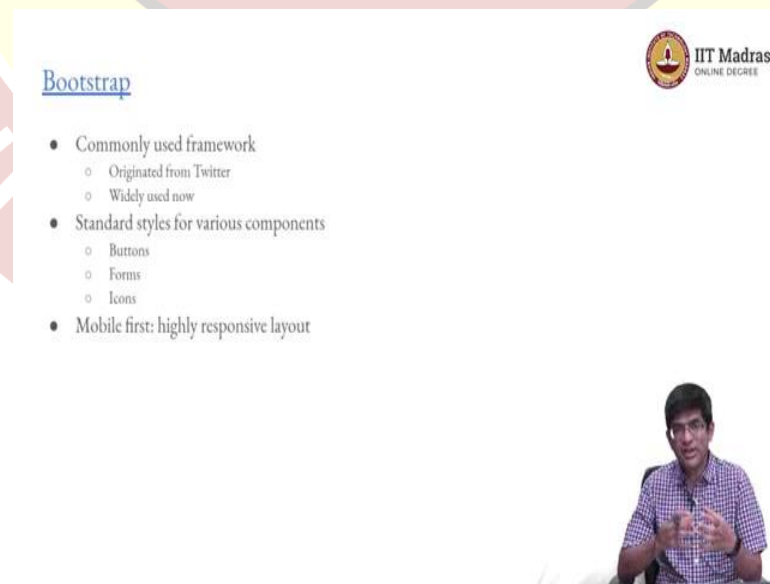
Now, having said that, can I go one step further? What happens if I go down to something really small like a phone? Now, on a phone even though I have shown it large over here, what I

actually have is the width is just probably three or four inches, whereas over here, it would be something like seven or eight inches at least. Within those three or four inches if I tried to take all of this and what I have on the left hand side and squeeze it down, it would still look very smart. Instead, it sort of adapts that. So, even something which, this text which was sort of flowing across two lines over here, now flows across multiple lines over here. The title and so on are also put into multiple lines without sort of sacrificing on the font size.

Now, of course, it does look like a lot of information has gone over here, but it is very easy to now scroll through this and pick out information or for that matter to use the buttons right on top, and pick the parts that you find most useful. So, this is basically what responsive design is all about. How is it enabled, because of the fact that we were able to separate out the HTML, the content from the styling, which was handled by CSS. And various sort of combinations of CSS are what we use in order to basically say, look, I can now just adapt to the size of the screen that I am looking at. And it will change the way it is displayed in such a way that it still looks good.

How do you decide that it looks good? Somebody has sat and sort of tried to tweak it so that the template that you are using adapts nicely to the new screen size and still looks readable on it. It is not that automatically CSS knows how to do it. But one person if they have done it once, that can now be used repeatedly by lots of other people.

(Refer Slide Time: 11:17)



Bootstrap

- Commonly used framework
 - Originated from Twitter
 - Widely used now
- Standard styles for various components
 - Buttons
 - Forms
 - Icons
- Mobile first: highly responsive layout

IIT Madras
ONLINE DEGREE

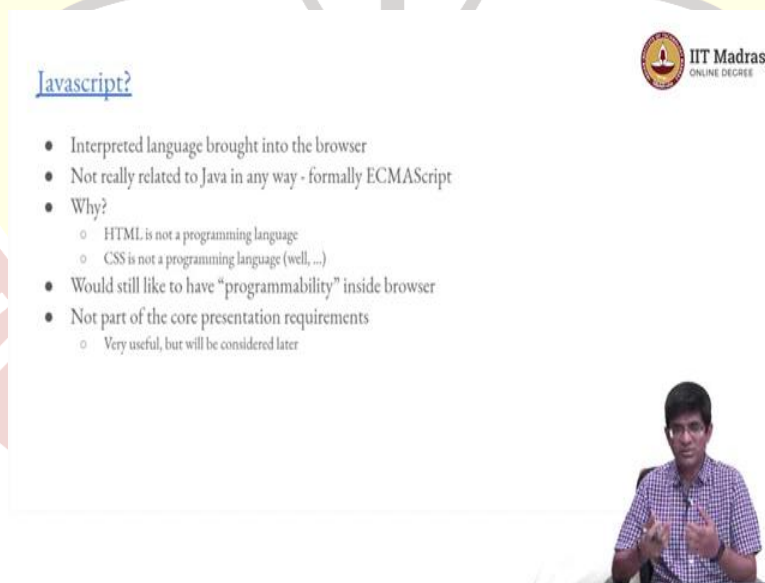
11

Now, this whole idea of reuse, the fact that CSS components can be reused by different people in order to get similar functionality or similar styling without much effort has led to a number of different frameworks for CSS. And one of the most commonly used ones is something called bootstrap. Bootstrap by itself is not really a programming language or a framework as such. It is more a framework specifically for CSS for style sheets. It originated from Twitter, but it is widely used all over the place now. It is open source.

And the nice thing about it is it defines very sort of aesthetic, aesthetically pleasing combinations of colors and layouts for things like buttons, tabs, navigation, the three line menu bar that you are probably familiar with and it is also designed right from the get go to being mobile first. Highly responsive layouts, Twitter is sort of most popular on mobile phones. Therefore, they made sense for them to focus on that. But it also does well on larger screens.

So, bootstrap is an example that can be used. But keep in mind that this is just one of several things. The whole idea over here is that the style sheets have made it possible to reuse material. This is one example of that being done.

(Refer Slide Time: 12:49)



JavaScript?

- Interpreted language brought into the browser
- Not really related to Java in any way - formally ECMAScript
- Why?
 - HTML is not a programming language
 - CSS is not a programming language (well, ...)
- Would still like to have "programmability" inside browser
- Not part of the core presentation requirements
 - Very useful, but will be considered later

IIT Madras
ONLINE DEGREE

Now, having come through all of this and the document object model, the APIs, the styling and all of that, one thing that I have not touched upon at all is JavaScript. What is the role of JavaScript? What is JavaScript? And what is its role in the web? It is an interpreted language that was basically brought into the browser. In some ways, it was brought in such that it was possible

to run some kinds of basic, non-trivial functionality, create some slightly more complex behaviors inside the browser that are not possible using HTML and CSS.

Now, why is this necessary, because although you may come across the terms HTML programming and CSS programming, neither HTML nor CSS is a programming language. You cannot really use it to control the behavior of a computer. If you give HTML there is something inside the computer which is already a program which interprets how to display it. If you give CSS, that will again modify it. But you are not, by writing the HTML code, you are not directly modifying the behavior of the computer.

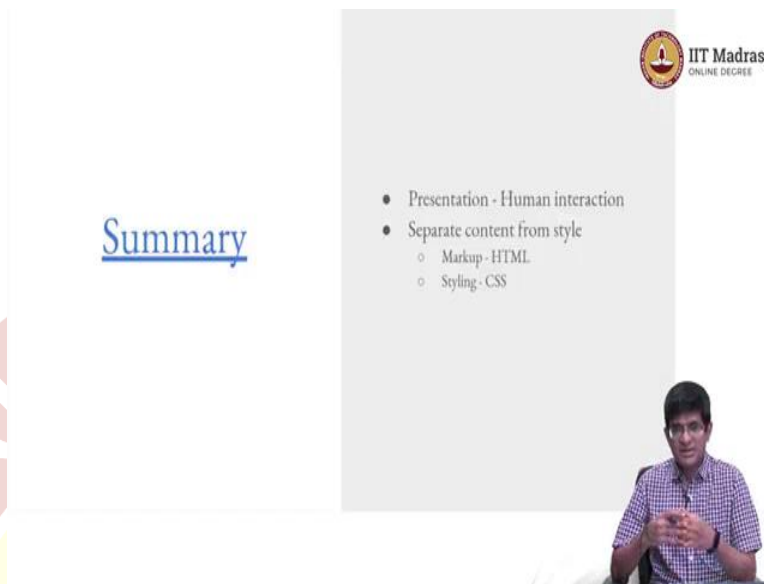
Now, this is a slightly tricky statement. The reason being that there are some specific set of conditions under which CSS can be shown to be what is called Turing complete, meaning that it can basically execute anything that any other computer can do. You can write a program, in other words, in CSS. But it requires some additional functionality such as user input and so on. The point is, it is not meant as a programming language. It is not meant to run for loops and take user input and do a bunch of other things just like that. It is meant for styling.

JavaScript, on the other hand, from the get go is meant for programming. It is meant to basically manipulate the DOM, manipulate the Document Object Model, either directly based on user input or without having any user interaction, but right from the beginning it basically says once a JavaScript starts executing, it is capable of modifying what is being shown on the screen. The original thing, why it came up was fairly simple tasks.

For example, when let us say, I am entering an email address, I can do the validation, is this a valid email address right there in the browser before even going to the server. So, certain kinds of things can be done automatically in the JavaScript, which sort of make the interaction between the user and the server a little bit more fluid. Nowadays, of course, JavaScript has got to the point where the entire page could be dynamically generated using JavaScript that has both good and bad aspects, which at some point, hopefully, we will get into those.

But the main thing is that as far as this app development course is concerned, this is, JavaScript is not going to be the primary aspect of what we are looking at. It is something which is useful, immensely useful, but we want to focus more on the basics, the core presentation layer and how do we create the functionality that we want from the server.

(Refer Slide Time: 16:13)



Summary

- Presentation - Human interaction
- Separate content from style
 - Markup - HTML
 - Styling - CSS

IIT Madras
ONLINE DEGREE

So, to summarize, this whole discussion has been about the presentation. In other words, the part that interacts with the user that interfaces with the user. And the main core idea that we wanted to bring out over here is this business of separating content from style. There is markup, which is typically done using HTML for web pages, versus styling, which in our case is handled using CSS, Cascading Style Sheets. And that logical separation between the two is something that is very important to understand and keep clearly in mind, because it has a direct impact on the design of applications, how they look, how they can be styled, how they can be developed or adapted moving forward.