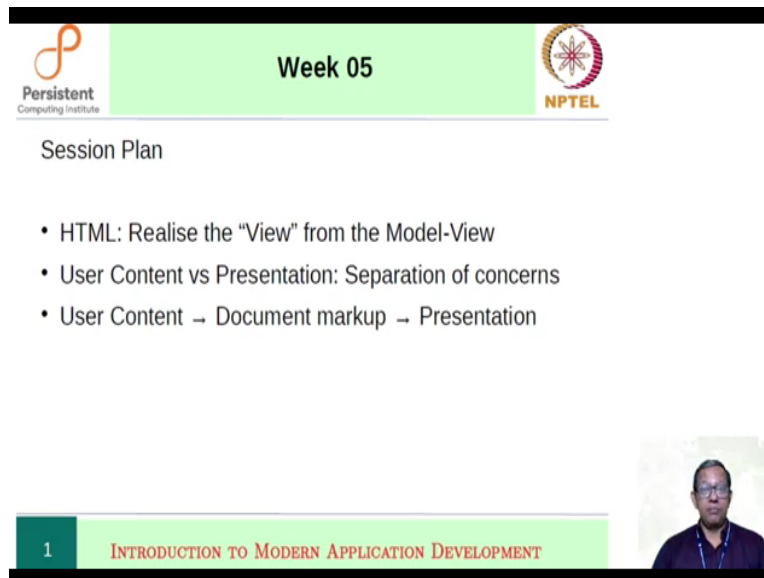


Introduction to Modern Application Development
Prof. Aamod Sane
FLAME University and Persistent Computing Institute
Abhijat Vichare
Persistent Computing Institute
Madhavan Mukund
Chennai Mathematical Institute

Lecture-17
Session 1-Part 1_Introduction to HTML and CSS

(Refer Slide Time: 00:11)




Week 05

Session Plan

- HTML: Realise the “View” from the Model-View
- User Content vs Presentation: Separation of concerns
- User Content → Document markup → Presentation


1 INTRODUCTION TO MODERN APPLICATION DEVELOPMENT




Hello, everyone, welcome to the fifth week of the course on modern applications development. We have come a long way since we began. We have the basic web application infrastructure ready with us. We started from a spreadsheet. We went on to the command line. We gradually modified the command line, so that it produced HTML output. Finally, we also added the ability to accept input using HTML; using the forms interface.

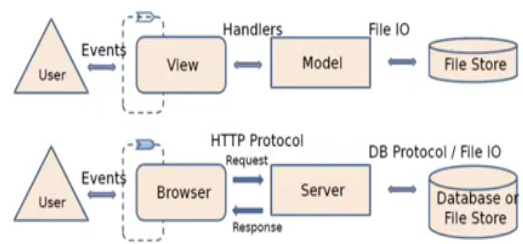
In this week, we will look at this view in a little more detail and understand the HTML and CSS operationally. Prof. Sane has discussed the conceptual aspects behind this idea. We will quickly review those ideas and go ahead with trying to understand the workings of actual HTML and CSS.

(Refer Slide Time: 01:22)




HTTP: Realising Model-View





2


INTRODUCTION TO MODERN APPLICATION DEVELOPMENT




In his sessions, Professor Sane has shown you this picture, where he contrasted the architecture of a graphical interface and architecture of a web system. The view and model separating these two aspects of a program has been an architectural innovation in some sense. In a graphical system, a part of the code is dedicated to the views and the other part is dedicated to the model; with handlers that communicate between them.

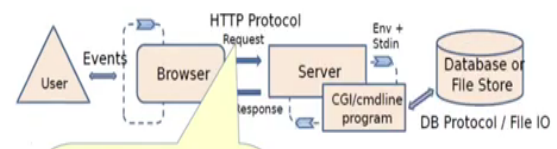
Very similarly in a web system, you have the HTTP protocol, as the glue, that binds the view aspect that is the responsibility of the browser and the model or any computation that is the responsibility of the server.

(Refer Slide Time: 02:28)




HTTP: Realising Model-View





3

INTRODUCTION TO MODERN APPLICATION DEVELOPMENT



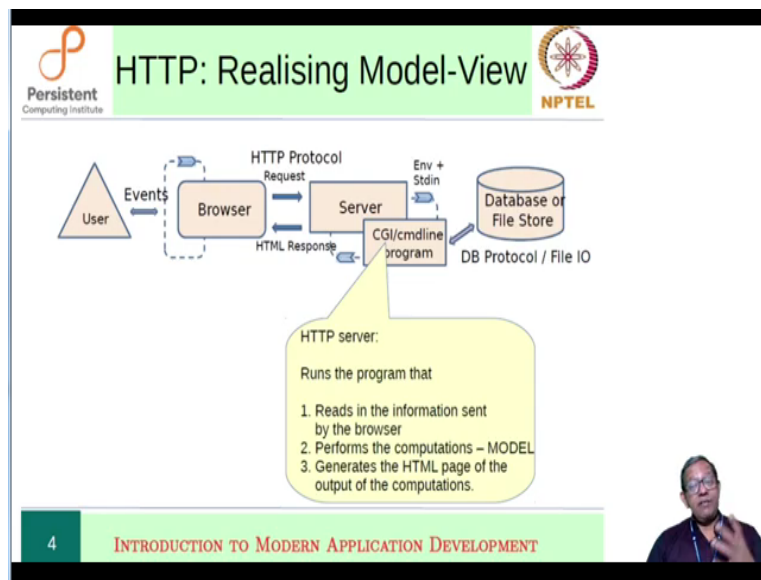
Form on the fairShare app:

Room mate types in the sequence of commands, and presses "Submit".

Browser collects the data, packages it as a "REQUEST", and sends it to HTTP server.

We saw that in action when the form interface of our FairShare app accepted a sequence of commands that otherwise were done on the standard input of your terminal. Your roommates typed in a sequence of commands and submit. This formed the request aspect of the HTTP protocol from the browser to the server. The browser collected the data that was there in the form, packaged it as a request and sent it over to the server.

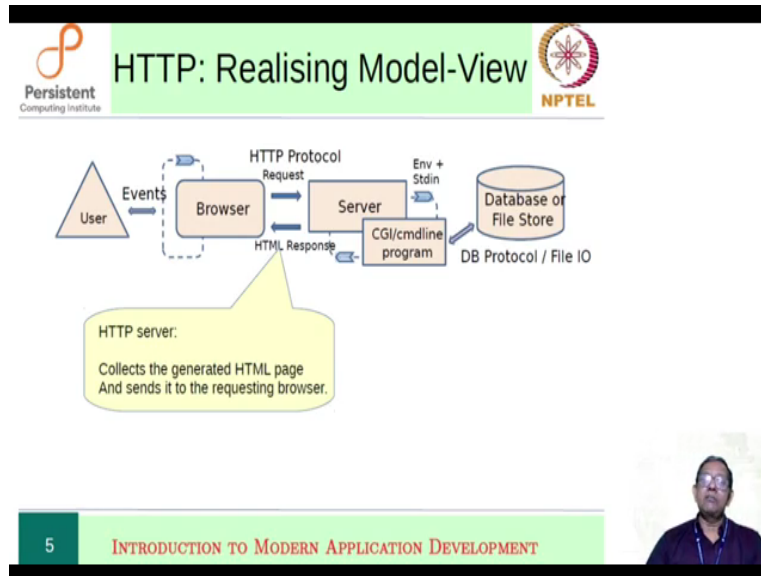
(Refer Slide Time: 03:11)



The server, on receiving that packet, ran a program that read the information that was sent by the browser, performed whatever computations were necessary. In our case, it was a FairShare application that actually responded to the commands. So, when registered command was received, our FairShare application performed the registration of the roommates. If expense command was received, then the expenses were recorded.

If the report was to be generated, it was done so and finally when it all ended, our FairShare application generated the HTML page that contained all this information as an HTML page.

(Refer Slide Time: 03:57)



It was this HTML page that was sent by the server back to the requesting browser. This is how the HTTP system realizes the model view aspect of the idea that professors Sane mentioned.

(Refer Slide Time: 04:19)

User Content & HTML: Together?

The INFORMATION our **users seek**: The FairShare System.

Warning: This is the HTTP version of the FairShare application with only HTML, and NOT CGI output.

Roommate Registrations

Registered Roommates are:

1. D
2. D

Number of Roommates: 2

Expenses record

1. D: 200.0

Report

1. D: -100.0

The Database

Number of Roommates	62
Number of Events	81
Number of Database entries	81


Event # 01 02 03 04

00 0.00 200.00 100.00 100.00


6 INTRODUCTION TO MODERN APPLICATION DEVELOPMENT

Let us begin our journey of trying to understand HTML. Let's start from; what the- user really seeks in the information?' On the right of your screen, you see the output of a FairShare system in a plain HTML form or the results of rendering a simple HTML page. Let us keep the HTML aside for a moment. From the user perspective, what really is the information that is really relevant to a user?

(Refer Slide Time: 05:09)



User Content & HTML: Together?



The INFORMATION our **users seek**: The FairShare System.

Welcome. This is the HTTP version of the FairShare application with only HTML, and NO CSS output.

Roommate Registrations

Registered Roommates are:

1. ID
2. ID

Number of Roommates: 2

Expenses record

1. ID: 200.0

Report

1. ID: 100.0


The Database

Number of Roommates	02
Number of Events	01
Number of Database entries	01


Event #	01	02	03
01	0.00 200.00	0.00 00 000.00	

A "Title" part that gives the name and some general information.


7
INTRODUCTION TO MODERN APPLICATION DEVELOPMENT



A title that gives a brief overview of what is the following information really about.
(Refer Slide Time: 05:19)



User Content & HTML: Together?



The INFORMATION our **users seek**: The FairShare System.

Welcome. This is the HTTP version of the FairShare application with only HTML, and NO CSS output.

Roommate Registrations

Registered Roommates are:

1. ID
2. ID

Number of Roommates: 2

Expenses record

1. ID: 200.0

Report

1. ID: 100.0


The Database

Number of Roommates	02
Number of Events	01
Number of Database entries	01


Event #	01	02	03
01	0.00 200.00	0.00 00 000.00	

A "Registration" section giving the details of the roommates registered.


8
INTRODUCTION TO MODERN APPLICATION DEVELOPMENT



We might think of a registration section that gives us details of the registration of the roommates.
In other words, which set of roommates does our current system know about.
(Refer Slide Time: 05:36)



User Content & HTML: Together?



The INFORMATION our **users seek**: The FairShare System.

Welcome. This is the HTTP version of the FairShare application with only HTML, and NO CSS output.

Roommate Registrations

Registered Roommates are:

1. 01
2. 02

Number of Roommates: 2

Expenses record

1. 02: 200.0

Report

1. 01: 100.0

The Database

Number of Roommates	02
Number of Events	01
Number of Database entries: 01	


Event # 01: 01 01 02

01	0.000 200.000 - 0.000 000 0.000 000
----	-------------------------------------


9
INTRODUCTION TO MODERN APPLICATION DEVELOPMENT

We would want to have a transcript record of the expenses that were done. These are pure expenses in the sense that what for every event, the amount that was paid by which roommate those statements, it is just a list of those statements.

(Refer Slide Time: 05:59)



User Content & HTML: Together?



The INFORMATION our **users seek**: The FairShare System.

Welcome. This is the HTTP version of the FairShare application with only HTML, and NO CSS output.

Roommate Registrations

Registered Roommates are:

1. 01
2. 02

Number of Roommates: 2

Expenses record

1. 02: 200.0

Report

1. 01: 100.0

The Database

Number of Roommates	02
Number of Events	01
Number of Database entries: 01	


Event # 01: 01 01 02

01	0.000 200.000 - 0.000 000 0.000 000
----	-------------------------------------


10
INTRODUCTION TO MODERN APPLICATION DEVELOPMENT

And finally, we would really want to have a report part where the report commands for whichever roommate that we needed, were given. And in response to the report command, our FairShare system gave information about any paybacks that were required.

(Refer Slide Time: 06:26)



User Content & HTML: Together?



The INFORMATION our **users seek**: The FairShare System.

Warning: This is the HTML version of the FairShare application with only HTML, and NO CSS output.

A "General Information" section that gives the other details like the contents of the database.

This is optional, though.

Roommate Registrations

Registered Roommates are:

1	01
2	02

Number of Roommates: 2

Expenses record

1 01 200.0

Report

1 01 100.0


The Database

Number of Roommates	02
Number of Events	01
Number of Database entries	01

Event #	01	02	03
01	0.00	200.00	0.00

11

INTRODUCTION TO MODERN APPLICATION DEVELOPMENT



It is possible that we might add some additional information. We might seek the actual database of the information that was contained in the transactions of the roommates. We could give some simple information like the number of roommates, the number of events and so forth. We could also list out the actual entire database of computations that were performed by our system. Let me reiterate the point: we are looking at the kind of information a user might really want.

We are not yet dealing with the question, how to represent this information using HTML. We are asking: what is it that the user would want as information? Registration section with registration information, a expenses section with a record of the expenses that were done and the reports section, which contain the reports for whichever friends the reports were requested for. We might possibly need more information optionally, in terms of general information, like the number of roommates or a detailed database of all the calculations performed.

(Refer Slide Time: 08:04)

User Content & HTML: Together?

The INFORMATION our users seek: The FairShare System.

How do we actually use HTML to display this user content?

Roommate Registrations

Expenses record

Report

The Database

12 INTRODUCTION TO MODERN APPLICATION DEVELOPMENT

Let us now go ahead and see how this required information by the user, which we will call as user content. How do we actually use HTML to display this user information? As usual, this would be a good point to take a little bit of a pause and recall what we have done, review what we have just said.

(Refer Slide Time: 08:39)

User Content Organization

What we need!

Title with Short description

The FairShare System.

Roommate Registrations

Expenses record


Report

The Database

13 INTRODUCTION TO MODERN APPLICATION DEVELOPMENT


As we said before, on your screen, you see the bear form of information that was generated by the FairShare application. Let us now mark parts of the application conceptually in various ways. We seek, for example, a title with a short description.

(Refer Slide Time: 09:07)



User Content Organization

What we need!



The FairShare System.

Welcome. This is the HTTP version of the FairShare application with only HTML, and NO CSS output.

Section Headings

Roommate Registrations

Registered Roommates are:

- 1. 01
- 2. 02

Number of Roommates: 2

Expenses record

1. 01: 200.0

Report

1. 01: 100.0

The Database

Number of Roommates: 02

Number of Events: 01


Number of Database entries: 01

Event # 01 02 03 04


01 0.00 200.00 0.00 0.00 0.00 0.00

14

INTRODUCTION TO MODERN APPLICATION DEVELOPMENT




There would be section headings to separate out individual pieces of information.
(Refer Slide Time: 09:18)



User Content Organization

What we need!



The FairShare System.

Welcome. This is the HTTP version of the FairShare application with only HTML, and NO CSS output.

Lists of various items

Roommate Registrations

Registered Roommates are:

- 1. 01
- 2. 02

Number of Roommates: 2

Expenses record

1. 01: 200.0

Report

1. 01: 100.0

The Database

Number of Roommates: 02

Number of Events: 01


Number of Database entries: 01

Event # 01 02 03 04


01 0.00 200.00 0.00 0.00 0.00 0.00

15

INTRODUCTION TO MODERN APPLICATION DEVELOPMENT




The pieces of information themselves could be lists of various items.
(Refer Slide Time: 09:24)



User Content Organization

What we need!



The FairShare System.

Welcome. This is the HTTP version of the FairShare application with only HTML, and NO CSS output.

Roommate Registrations

Registered Roommates are:

1. 01
2. 02

Number of Roommates: 2

Expenses record

1. 01	200.0
-------	-------

Report

1. 01	1000.0
-------	--------

The Database

Number of Roommates	02
Number of Events	01
Number of Database entries 01	


Event # 01	01	01	02	
00	0.00	200.00	400.00	000.00

Data Tables in various layouts

16
 INTRODUCTION TO MODERN APPLICATION DEVELOPMENT


They could also be tables of information. So, we are now aware, of what we seek as information that we want our users to see. We also see what the logical components of that organization are. We require a title, we require headings, and we require ability to list out some pieces of information. We require the ability to tabulate information and so on and so forth. It is exactly this aspect of the information that is marked by using the HTML system; HyperText Markup Language.

(Refer Slide Time: 10:16)



User Content Organization

What we need!



The FairShare System.

Welcome. This is the HTTP version of the FairShare application with only HTML, and NO CSS output.

Roommate Registrations

Registered Roommates are:

1. 01
2. 02

Number of Roommates: 2

Expenses record

1. 01	200.0
-------	-------

Report

1. 01	1000.0
-------	--------

The Database

Number of Roommates	02
Number of Events	01
Number of Database entries 01	

Event # 01	01	01	02	
00	0.00	200.00	400.00	000.00

Title with Short description

17
 INTRODUCTION TO MODERN APPLICATION DEVELOPMENT

(Refer Slide Time: 10:19)



User Content Organization

Marking **What** we need!



User Content Mark up using HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>The FairShare System.</title>
</head>
<body>
<h1>The FairShare System.</h1>
<p>Welcome. This is the HTTP version of the FairShare application with only HTML, and no CSS output.</p>
<h2>Roommate Registrations</h2>
<p>Registered Roommates are:</p>
<ol>
<li>1. 11</li>
<li>2. 12</li>
</ol>
<p>Number of Roommates: 2</p>
<h2>Expenses record</h2>
<p>1. 12: 200.0</p>
<h2>Report</h2>
<p>1. 11: 100.0</p>
<h2>The Database</h2>
<p>Number of Roommates: 42</p>
<p>Number of Events: 41</p>
<p>Number of Database entries: 81</p>
<p>Power # 11 12 11 12</p>
<p>00 0.00 200.00 0.00 0.00 0.00 0.00</p>

```

The FairShare System.

Welcome. This is the HTTP version of the FairShare application with only HTML, and no CSS output.

Roommate Registrations

Registered Roommates are:

1. 11
2. 12

Number of Roommates: 2

Expenses record

1. 12: 200.0

Report

1. 11: 100.0

The Database

Number of Roommates: 42
Number of Events: 41
Number of Database entries: 81

Power # 11 12 11 12
00 0.00 200.00 0.00 0.00 0.00 0.00

18
INTRODUCTION TO MODERN APPLICATION DEVELOPMENT

We now see the same information that was presented simply as ‘text’, we now see an HTML version of it. Although, it is a very small font and almost invisible, it is actually a complete HTML page. Let’s now focus on the parts of this page and study how HTML markup system has been used to organize the user content. Let me repeat we are going to reorganize the user content using HTML.

(Refer Slide Time: 11:03)



User Content Organization

Marking **What** we need!



User Content Mark up using HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>The FairShare System.</title>
</head>
<body>
<h1>The FairShare System.</h1>
<p>Welcome. This is the HTTP version of the FairShare application with only HTML, and no CSS output.</p>
<h2>Roommate Registrations</h2>
<p>Registered Roommates are:</p>
<ol>
<li>1. 11</li>
<li>2. 12</li>
</ol>
<p>Number of Roommates: 2</p>
<h2>Expenses record</h2>
<p>1. 12: 200.0</p>
<h2>Report</h2>
<p>1. 11: 100.0</p>
<h2>The Database</h2>
<p>Number of Roommates: 42</p>
<p>Number of Events: 41</p>
<p>Number of Database entries: 81</p>
<p>Power # 11 12 11 12</p>
<p>00 0.00 200.00 0.00 0.00 0.00 0.00</p>

```

The FairShare System.

Welcome. This is the HTTP version of the FairShare application with only HTML, and no CSS output.

Roommate Registrations

Registered Roommates are:

1. 11
2. 12

Number of Roommates: 2

Expenses record

1. 12: 200.0

Report

1. 11: 100.0

The Database

Number of Roommates: 42
Number of Events: 41
Number of Database entries: 81

Power # 11 12 11 12
00 0.00 200.00 0.00 0.00 0.00 0.00

19
INTRODUCTION TO MODERN APPLICATION DEVELOPMENT

Here is the initial part of the HTML system that you just saw. The arrows show you which part of the HTML correspond to what we, as users, are seeking and organizing. As users we seek a title and as organizers of the information, we want a part of the document to be explicitly marked

as title and a heading. We see the title tags, the one in red, and we see the heading of level h1 in the current system, as displayed right.

(Refer Slide Time: 11:59)

The slide is titled "User Content Organization Marking What we need!". It features the Persistent Computing Institute logo on the left and the NPTEL logo on the right. The main content is divided into two parts: HTML code on the left and a rendered output on the right.

HTML Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>The FairShare System.</title>
</head>
<body>
<div>The FairShare System.</div>Welcome. This is the HTTP
version of the FairShare application with only HTML, and
no CSS output.</div>
<div>Roommate Registrations</div>
<div>Registered Roommates are:</div>
<div>
<div>1 0</div>
<div>2 0</div>
</div>
<div>Number of Roommates: 2</div>
</div>
<div>Expenses record</div>
<div>Expenses record:</div>
<div>
<div>1 0</div>
<div>2 0</div>
</div>
<div>Report</div>
<div>Report:</div>
<div>
<div>1 0</div>
<div>2 0</div>
</div>
<div>The Database</div>
<div>The Database:</div>
<div>
<div>1 0</div>
<div>2 0</div>
</div>
</body>
</html>
```

Rendered Output:

The FairShare System.

Welcome. This is the HTTP version of the FairShare application with only HTML, and no CSS output.

Roommate Registrations

Registered Roommates are:

1	0
2	0

Number of Roommates: 2

Expenses record

Expenses record:

1	0
2	0

Report

Report:

1	0
2	0

The Database

The Database:

Number of Roommates	02
Number of Events	00
Number of Database entries	00

Event # 01 02 03 04

01 0.00 200.00 0.00 0.00 0.00 0.00


User Content Mark up using HTML

20 INTRODUCTION TO MODERN APPLICATION DEVELOPMENT

Heading level-2 roommate registrations as a heading, expenses record as a heading, report as a heading and the database as a heading of level-2 is used to separate out logical pieces of information. In the registrations part, we are going to list out the friends who formed the roommates here. In the expenses section, we are going to list out the set of expenses that this group of roommates has carried out and so on so forth.

What we are showing here is, how a simple text system is simply marking what part of the text is to be representing what part of the information that we want to organize. There is no statement about how the information is to be presented. For example, when we see a heading level-2 as roommate registrations over here, this is simply typed in as text by us. When this is displayed by the browser, it is displayed like this.


(Refer Slide Time: 13:38)



Persistent
Computing Institute

User Content Organization

Marking What we need!



NPTEL

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>The FairShare System.</title>
</head>
<body>
  <div>The FairShare System.</div>Welcome. This is the HTTP
  version of the FairShare application with only HTML, and
  no CSS output.</div>
  <div>Roommate Registrations</div>
  <div>Registered Roommates are:</div>
  <div>
    <ol>
      <li>f1</li>
      <li>f2</li>
    </ol>
  </div>
  <div>Number of Roommates: 2</div>
  <div>Expenses record</div>
  <div>
    <ol>
      <li>f2: 200.0</li>
    </ol>
  </div>
  <div>Report</div>
  <div>
    <ol>
      <li>f1: 100.0</li>
    </ol>
  </div>
  <div>The Database</div>
  <div>
    Number of Roommates: 02
    Number of Events: 01
    Number of Database entries: 01
  </div>
  <div>
    f1: 100.0
    f2: 200.0
    f3: 300.0
  </div>
</body>
  </html>

```

The FairShare System.

Welcome. This is the HTTP version of the FairShare application with only HTML, and no CSS output.

Roommate Registrations

Registered Roommates are:

- f1
- f2

Number of Roommates: 2

Expenses record

1. f2: 200.0

Report

1. f1: 100.0


The Database

Number of Roommates	02
Number of Events	01
Number of Database entries	01

Event # f1 f2 f3

100.0	200.0	300.0
-------	-------	-------

User Content Mark up using HTML



21

INTRODUCTION TO MODERN APPLICATION DEVELOPMENT

We now see how lists of information are created and presented, as creators we only create a list of information. Here is a list of here is a tag; HTML tag, that is used to present ordered lists. By ordered we mean that they are enumerated: 1 2 3 4 on and so on so forth. An ordered list is tagged between OL and forward slash OL. The members of the list are tagged between list item LI and forward slash LI to denote the end of the current list item in this case it is f1.

The next list item is f2 which is again enclosed in the tags LI list I2 and forward slash LI. In expenses record, we again have an ordered list OL and forward slash OL with list items f2 200 between LI and forward slash LI. This is how we write ordered list in HTML. When we mark up, we are simply writing text documents with a very standardized system that marks the text for whatever the eventual purpose of organization is.

We wish to organize this as a list item for expenses section. Therefore it is here is the organization of the list and it is coming under the heading h2 for expenses.

(Refer Slide Time: 15:38)



Persistent
Computing Institute

User Content Organization

Marking What we need!



NPTEL

```

<!--The Database-->
<TABLE>
<TR>
<TD>Number of Roommates</TD>
<TD>2</TD>
</TR>
<TR>
<TD>Number of Events</TD>
<TD>4</TD>
</TR>
<TR>
<TD>Number of Database entries</TD>
<TD>10</TD>
</TR>
</TABLE>
<!--Event #-->
<TABLE>
<TR>
<TD>Event #</TD>
<TD>1</TD>
<TD>2</TD>
<TD>3</TD>
<TD>4</TD>
</TR>
<TR>
<TD>Room</TD>
<TD>100</TD>
<TD>200</TD>
<TD>300</TD>
<TD>400</TD>
</TR>
</TABLE>
</BODY>
</HTML>

```

The FairShare System.

Warning: This is the 1977 version of the FairShare application with only HTML, and NO CSS output.

Roommate Registrations

Registered Roommates are:

1. 15
2. 12

Number of Roommates: 2

Expenses report

1. 12: 200.0

Report

1. 15: 100.0

User Content Mark up using HTML

The Database

Number of Roommates:	02
Number of Events:	04
Number of Database entries:	01

Event #	12	11	12
01	0.000 200.00	0.000 200.00	



22

INTRODUCTION TO MODERN APPLICATION DEVELOPMENT



Here is the later part of the same webpage, HTML piece of code. Again, you see the database as the second level heading.



Persistent
Computing Institute

User Content Organization

Marking **What** we need!



NPTEL

The FairShare System.

Welcome. This is the 11777 version of the FairShare application with only HTML, and NO C30 output.

Roommate Registrations

Registered Roommates are:

1.05
2.02

Number of Roommates 2

Expenses report

1.01 200.0

Report

1.01 -100.0

The Database

Number of Roommates 02

Number of Events 01

Number of Database entries 01

Event #	01	02	03
01	0.00	200.00	-100.00

```

<?xml version="1.0"?>
<table>
  <tr>
    <td>Number of Roommates</td>
    <td>02</td>
  </tr>
  <tr>
    <td>Number of Events</td>
    <td>01</td>
  </tr>
  <tr>
    <td>Number of Database entries</td>
    <td>01</td>
  </tr>
</table>
<table>
  <thead>
    <tr>
      <th>Event #</th>
      <th>F1</th>
      <th>F2</th>
      <th>F3</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>01</td>
      <td>0.00</td>
      <td>200.00</td>
      <td>-100.00</td>
    </tr>
  </tbody>
</table>
</xml>
  
```

User Content Mark up using HTML

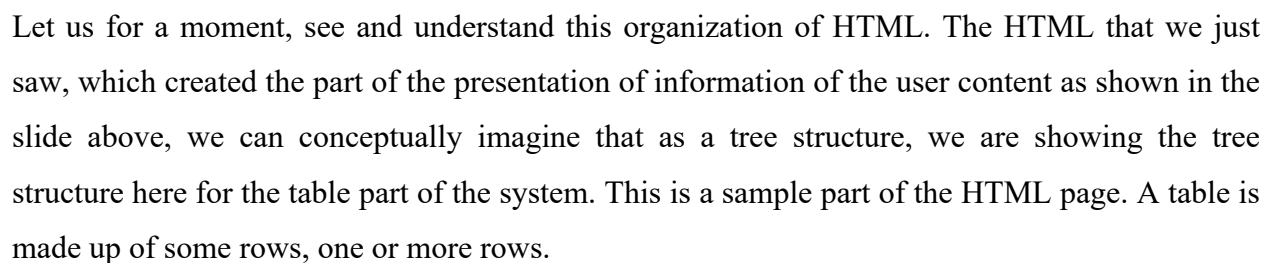
23

INTRODUCTION TO MODERN APPLICATION DEVELOPMENT



And we now see tables. Just like lists we now have tables which will create a table of information and values or information and records of values. It is possible to write this page by hand. In other words, you could simply open up your editor and start writing the page as we just saw. We start with this doc HTML, HTML head, body, h1, h2, p for paragraphs, ordered list, list items and so on so forth, we can just go and typing.

(Refer Slide Time: 17:08)



And that row had the first data item as the event number second data item as the information for friend f1 roommate f1. The third data item was information for roommate f2 and so on. This structure of the HTML document is called as a ‘document object model’ or the DOM. And the tree that result is called as a DOM tree. This conceptual view of an HTML page is extremely useful.

In fact, it corresponds to a very object oriented model of a web page. This object oriented model of a web page will be extremely useful for us in the future. A few sessions later, we will use actually use this particular document object model idea, much more usefully. However, at this point, it is also not as bad. It is something it is required for us to understand how HTML is going to be represented and its display will be done by the browser.

(Refer Slide Time: 20:00)

The slide, titled "Marking: An Overview", features the Persistent Computing Institute and NPTEL logos. It displays the structure of an HTML document with the following code snippets and annotations:

```
<!DOCTYPE html>
<HTML lang="en">
  <HEAD>
    <TITLE> ... </TITLE>
  </HEAD>
  <BODY>
    <H1>...</H1>
    <OL>
      <LI> ... </LI>
      ...
    </OL>
    ...
  </BODY>
</HTML>
```

Annotations on the right side of the slide include:

- See: Mozilla Developers - HTML Introduction
- See: The W3 HTML 5.x Specification
- The W3 Schools: HTML

The slide number 25 and the course title "INTRODUCTION TO MODERN APPLICATION DEVELOPMENT" are visible at the bottom. A small icon of a person at a computer is also present.

Here is a quick overview of the HTML page system. We are going to be compliant with HTML version 5. It starts with a DOCTYPE HTML as a declaration and the entire HTML page is enclosed between HTML and forward slash HTML. And the entire page is enclosed between this one specifies the language of the page. This is short for English. We are going to present this page in English. If you were to present it in Japanese, you would use the corresponding short.

An HTML page has sub parts the head which has the title, there could be other things that are also included in the head. In fact, those other things that are included in the head are very, very useful for search engines. Apart from the head, the HTML also has a body which is enclosed between the tags body and forward slash body. Among the many things that are there are heading at various levels, h1 that we just saw.

Ordered lists, list items and so on and so forth. Whatever you need to put, there is a whatever information you need to display, there is a way of hierarchically presenting it. We also have some links shown over here which you can use for more detailed study or as a reference for the various HTML tags and their uses. In fact, the most authoritative version would be in the W3 pages, this is an organization that defines the standards for the web of various kinds including HTML, CSS and so on and so forth.

There are a number of online tutorials also available one of them that is the one that we have mentioned over here, which you can use to know which are the other tags and how to use them. If one looks at it, they are fairly logical h1 represents heading level one. There are six such heading levels available in HTML that are defined. You have the abilities to present ordered list using OL. If the lists are unordered then they are represented using a tag UL for unordered lists.

Whether it is an ordered or an unordered list will always have a list item. All such kinds of detailed information can be obtained here from the original HTML specification site or more incrementally using various online tutorials like the W3 schools. There are many others that are also available. Before we leave, let us have a look at a web page. **(Video Starts: 23:35)** Here is a web page that is generated by one of our FairShare programs.

Let me take you step by step through this entire page. The first line over here is a comment. Comments in HTML are enclosed between this mark and this ending mark. Whatever is there between this for example, this part is command and is ignored by the browser. A web page starts with this declaration of the type of the document DOCTYPE. As I said, HTML pages are enclosed between 2 tags HTML and if I go at the end of the page, we will have a forward slash HTML.

The closing tag for every HTML there is a forward slash HTML closing tag. Here is the head, which has a title. The title is the FairShare system full stop and is enclosed between the title and the forward slash title. After the title we have the body of the page. The body of the page is composed of a heading at level 1. Then there is general description, a short one.

This tag 'br' is a line break. It is this 'br' tag that is used by the browser to go to the next line. This means that in your HTML page, you can split the line anywhere you want for your own convenience. And the browser is not going to use that line break that you have given. Hence, you can write the information of your web page in any way that is convenient to you.

Again, this is a command. Note that the command is used to just mark the start of the header and end of the header. And this is something that is useful to people who are going to write web pages. Here is a command that indicates that the HTML registration section part of the web page is starting. It starts with the roommate registrations as the header at level 2, therefore it is h2. And the header is as usual enclosed between two markers: h2 and forward slash h2.

A p tag starts a paragraph which, of course, will involve a new line as displayed on the web browser. Notice that, I start the ordered list on the same line as this part of this information that I have written. Recall that, the browser is not going to worry about the line breaks that we introduce. Whenever an ordered list is to be started, it always is to be displayed on the next line with proper enumeration: 1, 2, 3, 4 or A, B, C, D whatever may be.

Here are the list items. In this particular page, unlike the one that we saw in these lines, we have 4 roommates. As of now, no expenses have been recorded and no report has been requested. Therefore, the expenses section is marked by header at level 2. There is a statement that no expenses have been recorded. And a line break has been introduced in the HTML displayed version. The 'hr' tag creates a horizontal run line, the so called horizontal rule on the web page.

The registration section ends with a line and then the expenses section begins. It means as far as the user is concerned, the user will see a registration section, a line and an expenses section. It will be more neat than what we had presented the plan is the very simple information that we had presented on our slides. This will be a little bit better. Again, the report section enclosed using the h2 heading level 2 tag.

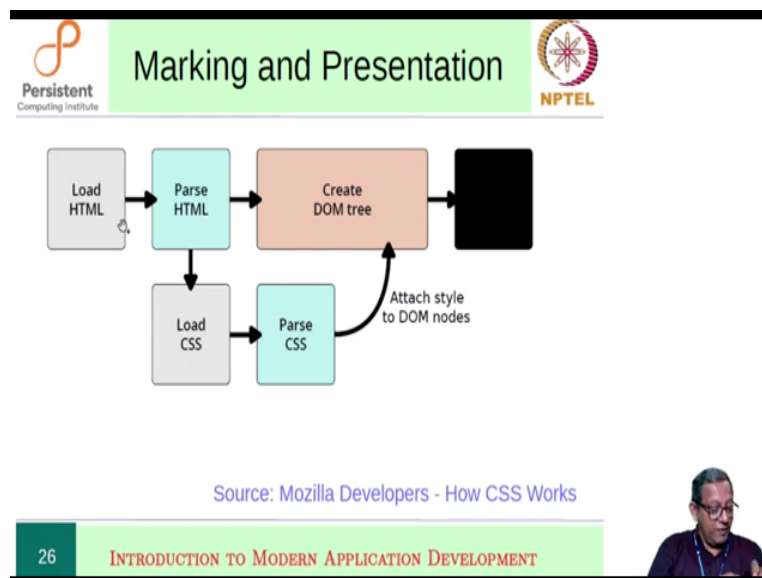
A line break here then on the new line, a horizontal line and fine and the next section; database information that has the table. We will ignore this class for a moment. And we will take care of that when we work with CSS that is cascading style sheets. As we saw in the document object

model side, a table is defined by a table row that table row has some data. The first column of table data contains the information file name.

Once that first column is finished using the forward slash td tag. Then, the next column of table data begins that has this piece of information and the column ends with a forward slash td. But there are only two columns. Therefore, the first row is now over and you go to the second row. The next row has one column with this information and the second column and so on and so forth.

The second-row finishes, the third row begins, the third-row finishes and so on and so forth. The table ends with a forward slash table and then we write a horizontal rule to separate it from the next table. The next table is marked with a heading at level 3. It says that right now the only thing that has been done is registrations, therefore, it is a fresh start and there is no other information in the database.

We have organized the user's information that the user seeks; the content. Thought(think) about how that organization is to be marked and used HTML to actually mark it. It will be good to try out some web pages for yourself and see how they appear on your browser. **(Video Ends: 32:37)**
(Refer Slide Time: 32:39)



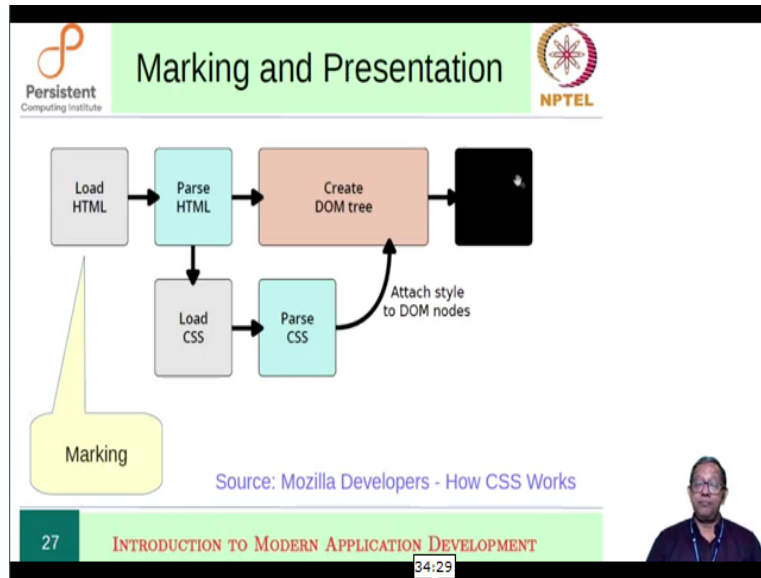
Having marked our information that we want to present to the user. Let us now look at the next part of the story, how to actually define the presentation. To start with, let us have a look at this conceptual picture of how an HTML page is rendered on by the browser. This picture is taken from the Mozilla developers pages which describes how CSS works.

This picture is from the point of view of a browser. A browser loads the HTML. By load we mean, the server it has connected to the server and whatever the server has sent as an HTML page, that page is loaded by the browser in its memory. It is parsed in the sense you can imagine that it constructs a document tree of that particular page that it has received. It creates a document and it displays over here. The black box on the right side is your screen.

The HTML that we saw was a very unadorned simple HTML. It only indicated what part of the information is to be marked. We had designed that information layout. We thought about what is the information that the user seeks. Having recognized that information we designed it like the first there would be a registration section, there would be an expenses section, there would be a report section, there would be optional databases information. Having organized in our minds the layout of the information we then used HTML to mark that layout.

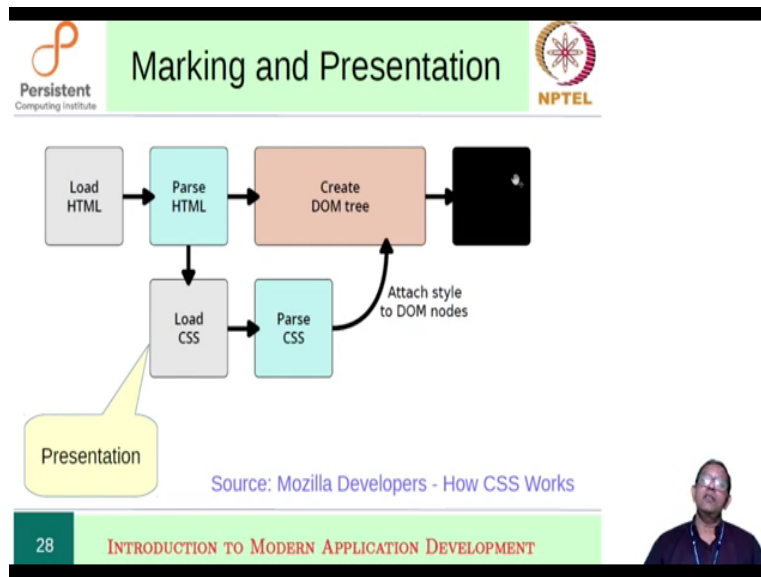
In order to perform presentation and to actually present that marked information, we can additionally take the support of the CSS; Cascading Style Sheets. Just like the HTML is loaded, our browsers will also load the CSS and parse it. The information in the CSS the style sheets will then be used to attach the various pieces of style to the objects in the DOM tree. And then they will be displayed.

(Refer Slide Time: 35:39)



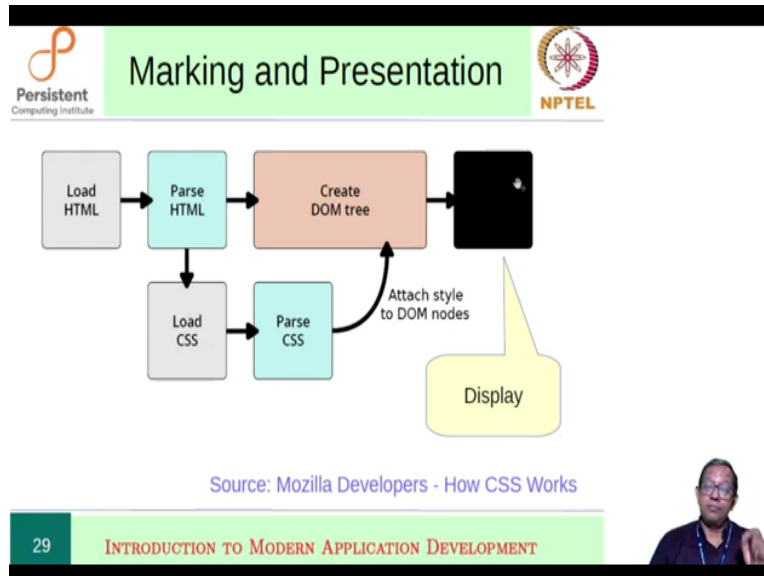
So, the HTML will be used to mark.

(Refer Slide Time: 35:42)



The CSS will be used to define the details of the presentation.

(Refer Slide Time: 35:47)



And the display is where both of these will come together and you will see a much better web page than the simple plain one that we had created. Let's try to understand this process of actually performing the presentation.

(Refer Slide Time: 36:10)

Presentation: CSS Basics

- Two basic specifications
 - Document Structural Layout attributes
 - Markup Layout attributes
- Markup layout attributes
 - e.g. font family, colors, sizes etc.
 - Format:


```
markup-selector : {
  attribute: value;
  ...
}
```
- Document layout attributes
 - divide HTML into groups called "classes"
 - Specify region over which the style *spans*.
 - Format:


```
class-selector : {
  attribute: value;
  ...
}
```

Source: Mozilla Developers - How CSS Works

30 INTRODUCTION TO MODERN APPLICATION DEVELOPMENT

What do we really do when we want to present? At this point, we are not going to concern ourselves with the markup system that is there. Our only concern is really with whatever information comes from the markup, how do we present it. There are two basic specifications that we really need to worry about. Given a document we would like to present regions of the document in separate ways.

For example, we would want to treat all the HTML that is a part of the registration section, we will want to treat them separately. For example; we might want them to have a different background color. We might want to actually separate sections of the information and display them in different ways. This sectioning is related to the entire document, we would want to group a bunch of HTML tags together.

That is what we would like to treat them as, we would like to divide the document into various parts. The other specification that we need to worry about is for individual HTML tags. We would want for example, the tables to be presented differently. After all, they present different kinds of information. One table is just presenting the values of some people of information, the other one is presenting the results of computation.

We might want to present these two tables differently. The markup only gives the logical content of the table. It only says that in row 1, column 1, this is the information row 2, column 2, that is the information, row 3, column 1, whatever else is the information and so on. It does not specify how the information is to be presented or how the table the table rows and the table columns are to be presented.

Perhaps, we want to have a different background color for the first column and a different font for the second column. So, in general, there will be two kinds of specifications that we need to really worry about. One is the markup layout: given an HTML markup tag, how is it to be presented. We might want to worry about attributes of presentation, like the font family, the colors of the text, the sizes of the font.

And many such attributes of presenting that given markup. We have already seen various markups. HTML forward slash HTML is one markup. Body slash forward slash body is another markup. Ordered list forward slash ordered list is another pair of markups. The general format of defining the presentation or specifying how a particular markup is to appear is as follows. You write the markup selector, for example; h1, h2, ordered list OL, whatever.

And separated by a colon and then within the braces for that markup, whatever attributes, you want to present and the value; this list is all that you really write. So, for example, one might write as `h1- colon - brace, font family; whatever font you want, font size; whatever font you want, and so on and close it`. This is an instruction to the browser that in the incoming HTML, wherever you find `h1`, it is to be displayed using the font that is given here, add the size that is given here and so on so forth.

This takes care of the individual markups. We will shortly see an example of CSS, but before that, let us look at the other kind of a part where we are concerned with dividing the document into various parts according to their logical organization of the information, and that is something that we know not the machine. We would like to divide the group of HTML tags into various subgroups. This division is specified by the `div` tag of HTML.

The `div` tag of HTML does not really mark text. It instead actually divides regions the HTML into various logical parts that we as designers create, you could divide the HTML into groups called classes or you could specify regions over which the given specification spans. There are ways of organizing the HTML page and we have just looked at using `div` and we have just shown how to organize HTML into groups using classes of the `div` tag.

The CSS specification format then is just like we had a markup selector, we have a class selector and again, attribute and values. This is the most basic way in which presentation is controlled or presentation is defined. As usual, we have provided some links which you can use to see how CSS works.

(Refer Slide Time: 43:15)

Persistent Computing Institute

Presentation: CSS Example

NPTEL

```
html {  
  background-color: #CCCCCC; /* RGB, each of two hex digits */  
  font-family: Arial;  
}  
  
body {  
  margin-left: 2em;  
  text-align: left;  
}  
  
h1 {  
  font-size: 150%;  
  font-style: bold;  
}
```

31 INTRODUCTION TO MODERN APPLICATION DEVELOPMENT

Here is an example: This example actually just illustrates how markup tags are to be presented. The HTML tag, there is a colon here that is missing in this example. In fact, throughout this entire slide, HTML colon, open brace, the background color has a value has a value hash CCCCCC. This is in the RGB format each are of 2 hex digits. So, this is red. This is green and this is blue. All of them have been mixed in equal proportion.

This actually will give us a gray colored background. The entire HTML page is to be rendered in a gray background. This is what this particular specification sees. It also says that the font to be used is Arial. One could use a variety of fonts that are available. The body of the page: body, colon-open brace and close brace within this, you have a left margin of two em's, it is actually two m's. This roughly corresponds to the width of the small m character.

And when we say two m, it means two small m's, that the from the left of the page, a margin of two m's has to be kept and the text is to be presented left aligned in the entire body. Note that the body will have the background of HTML because HTML is the main tag which defined the gray background for itself. And that information will spill over into the body unless the body also says that it wants its own background color.

The h1 tag, again there is a colon over here the h1 tag, says that the size of the font is to be increased 150%. This means that the header level 1 will be presented by the user's browser using

a font size that is 150% more than what is the default size of the browser. Moreover, it says that the font style is bold. So, the browser will present the user a bold, bigger font. For heading level 1, only. You can go on specifying the details of every tag in your CSS system.

(Refer Slide Time: 46:25)

Persistent Computing Institute NPTEL

Presentation: CSS Example

```
html {  
  background-color: #CCCCCC; /* RGB, each of two hex digits */  
  font-family: Arial;  
}  
  
body {  
  margin-left: 2em;  
  text-align: left;  
}  
  
h1 {  
  font-size: 150%;  
  font-style: bold;  
}
```

HTML Markup tags
The layout parameters of these tags are specified.

32 INTRODUCTION TO MODERN APPLICATION DEVELOPMENT

The HTML markup tags are shown.

(Refer Slide Time: 46:33)

Persistent Computing Institute NPTEL

Presentation: CSS Example


```
html {  
  background-color: #CCCCCC; /* RGB, each of two hex digits */  
  font-family: Arial;  
}  
  
body {  
  margin-left: 2em;  
  text-align: left;  
}  
  
h1 {  
  font-size: 150%;  
  font-style: bold;  
}
```

Style attributes and their values are listed for each tag.


33 INTRODUCTION TO MODERN APPLICATION DEVELOPMENT

The attributes, names of the attributes are shown using red arrows and the values are shown using green arrows.

(Refer Slide Time: 46:47)



Markup PLUS Presentation



```

<!DOCTYPE html>
<HTML lang="en">
<HEAD>
  <TITLE> ... </TITLE>
  <STYLE type="text/css" media="screen">
  </STYLE>
</HEAD>
<BODY>
  <H1>...</H1>
  <OL>
    <LI> ... </LI>
    ...
  </OL>
  ...
</BODY>
</HTML>

```

```

html {
  background-color: #CCCCCC;
  font-family: Arial;
}

body {
  margin-left: 2em;
  text-align: left;
}


h1 {
  font-size: 150%;
  font-style: bold;
}

```

34


INTRODUCTION TO MODERN APPLICATION DEVELOPMENT

46:38




How do we include this information in the web page? We are going to show two different styles : One is by using the style tag in your HTML page, within the style and forward slash style, the entire CSS that is the stylesheet specification is are written. We just type it in there. This is called as in-lining the styles. The styles specification how to present the information is made a part of the HTML page itself. This is called in-lining.

(Refer Slide Time: 47:36)



Markup PLUS Presentation



```

<!DOCTYPE html>
<HTML lang="en">
<HEAD>
  <TITLE> ... </TITLE>
  <LINK rel="stylesheet"
        href="/css/demo-css.css"
        type="text/css" />
</HEAD>
<BODY>
  <H1>...</H1>
  <OL>
    <LI> ... </LI>
    ...
  </OL>
  ...
</BODY>
</HTML>

```

Save as: css/demo-css.css

```

html {
  background-color: #CCCCCC;
  font-family: Arial;
}


body {
  margin-left: 2em;
  text-align: left;
}

h1 {
  font-size: 150%;
  font-style: bold;
}

```

35

INTRODUCTION TO MODERN APPLICATION DEVELOPMENT



The other approach is to save the style specifications into a file, in this case, we are showing we are saying that that file has been saved into a file called as demo dash CSS, dot CSS. And in the HTML page, we give a link or reference to that particular file. This allows us to separate the

presentation information in a separate file and the in markup information in a separate file. And they are linked together using the link tag in HTML.

(Refer Slide Time: 48:31)

The slide is titled "Markup PLUS Presentation" and features logos for "Persistent Computing Institute" and "NPTEL". It displays HTML and CSS code. The HTML code includes a link to a stylesheet. A callout box explains that multiple style sheet files can be linked and that style specifications cascade. The CSS code defines styles for the html and body elements.

```
<!DOCTYPE html>
<HTML lang="en">
<HEAD>
<TITLE> ... </TITLE>
<LINK rel="stylesheet" href="/css/demo-css.css">
</HEAD>
<BODY>
</BODY>
</HTML>
```

Save as: css/demo-css.css

```
html {
background-color: #CCCCC;
font-family: Arial;
}

body {
margin-left: 2em;
text-align: left;
}

h1 {
font-size: 150%;
font-style: bold;
}
```

Multiple style sheet files can be linked.
They can be from different sites too.
Style specifications cascade!
Hence the name!

36 INTRODUCTION TO MODERN APPLICATION DEVELOPMENT 49:11

This is interesting because you can now have multiple style sheets linked into the same page. The styles can be from different sites, too, because the specification of the stylesheet simply uses a href attribute and the style specifications will cascade. Recall what we said: whenever the body is rendered, the attributes of the higher level tag cascade information into that particular tag too. If we have a sequence of style sheets, then the information from each file will aggregate together until the final sheet is finished.

In other words, the specification of how to present from file 1 will cascade into the presentation information in file 2, which is following that. That is why the name is cascading style sheets or CSS for short.

(Refer Slide Time: 49:48)

The slide is titled "Presentation: CSS Example" and features the logos of the Persistent Computing Institute and NPTEL. It displays CSS code for the `html`, `body`, and `h1` elements. The `html` block sets the background color to `#CCCCCC` and the font family to `Arial`. The `body` block sets the left margin to `2em` and text alignment to `left`. The `h1` block sets the font size to `150%` and font style to `bold`. Three callout boxes provide additional resources: "W3: The CSS Standards specifications and reference", "CSS Zen Garden: Demos of CSS", and "The W3 Schools: CSS" with a URL. A small icon of a person at a computer is also present. The slide number "37" and the course title "INTRODUCTION TO MODERN APPLICATION DEVELOPMENT" are in the footer, along with a timestamp of "50:20".

```
html {
  background-color: #CCCCCC; /* RGB, each of two hex digits */
  font-family: Arial;
}

body {
  margin-left: 2em;
  text-align: left;
}

h1 {
  font-size: 150%;
  font-style: bold;
}
```

W3: The CSS Standards specifications and reference

CSS Zen Garden: Demos of CSS

The W3 Schools: CSS
<https://www.w3schools.com/css/default.asp>

37 INTRODUCTION TO MODERN APPLICATION DEVELOPMENT 50:20

This is again a good point to pause and review the material. We have also given some links to further information. The W3 of course, is the most authoritative site, which contains information about CSS standards. Again, there are a number of tutorials available on the internet, W3 schools is one of them. We also suggest have a look at sites the like the CSS Zen Garden, which demonstrates the power of CSS.

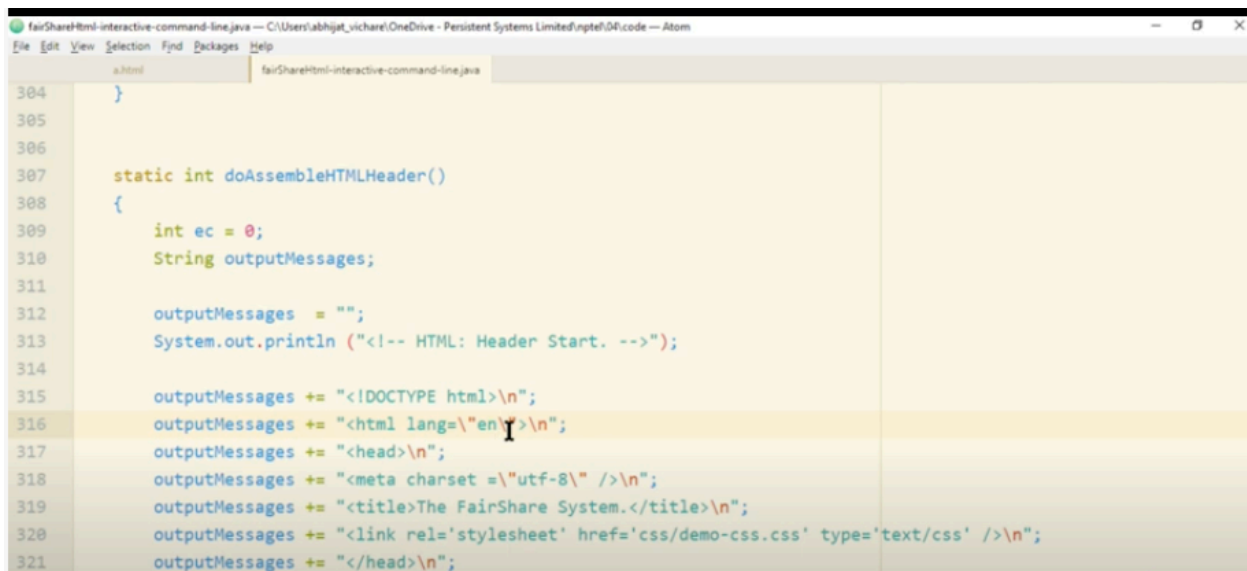
They will show you the same web page, same HTML markup rendered using different style sheets. And you can observe how the same markup can be rendered differently by using different style sheets and the changes are dramatic. **(Video Starts: 50:57)** Having looked at how HTML and CSS Work together; let us finish the final part of the puzzle. How will our program create such a thing?

Here is our FairShare application; the usual interactive one, that generates output in HTML. If you recall, the major computation part, the model part of the story was actually done by this method `doSetup`. Let us revisit that. But before we revisit, observed that after the computations are done, our Java program has been instructed to assemble the entire HTML page. Let us recall, `'doSetup'`. The `doSetup` fellow is quite interesting.

It reads the standard input for the various commands. Here is a Boolean assemble HTML: This will be true if the incoming mode of the command is end otherwise it is false. This means that a

web page will not be created until the user gives the end command in the entire transcript of commands. Only after the end command is received, will the webpage be generated. As usual, our command line program will perform the various tasks of registration, recording expenses, computing the reports, computing the FairShare and then reporting all of that will be done as usual.

Having finished all the registration expenses and reporting functions, let's assume that the user has given the end command, which means that a simple HTML would be true. So, at the end, when our program receives the end command, it will start assembling the HTML page. How does it do that? It is just a sequence of methods which assembles the HTML header the registration in information, the expenses information, the report information; it shows the database and finally does the tail part of the HTML.

A screenshot of the Atom code editor. The top bar shows the file path: C:\Users\abhiyot_richard\OneDrive - Persistent Systems Limited\ngtel\DF\code - Atom. The editor is open to a file named fairShare.html-interactive-command-line.java. The code is in Java and shows a static method doAssembleHTMLHeader(). The method starts with a block comment indicating it's for HTML header assembly. It initializes an integer variable 'ec' to 0 and a String variable 'outputMessages'. It then sets 'outputMessages' to an empty string and prints a comment. The core of the method is a series of concatenations using the '+=' operator to build the HTML header string. The lines shown are: 304: closing brace for a previous block; 305: empty line; 306: empty line; 307: static int doAssembleHTMLHeader(); 308: { 309: int ec = 0; 310: String outputMessages; 311: 312: outputMessages = ""; 313: System.out.println ("<!-- HTML: Header Start. -->"); 314: 315: outputMessages += "<!DOCTYPE html>\n"; 316: outputMessages += "<html lang='en'>\n"; 317: outputMessages += "<head>\n"; 318: outputMessages += "<meta charset = 'utf-8' />\n"; 319: outputMessages += "<title>The FairShare System.</title>\n"; 320: outputMessages += "<link rel='stylesheet' href='css/demo-css.css' type='text/css' />\n"; 321: outputMessages += "</head>\n";

```
304 }
305
306
307 static int doAssembleHTMLHeader()
308 {
309     int ec = 0;
310     String outputMessages;
311
312     outputMessages = "";
313     System.out.println ("<!-- HTML: Header Start. -->");
314
315     outputMessages += "<!DOCTYPE html>\n";
316     outputMessages += "<html lang='en'>\n";
317     outputMessages += "<head>\n";
318     outputMessages += "<meta charset = 'utf-8' />\n";
319     outputMessages += "<title>The FairShare System.</title>\n";
320     outputMessages += "<link rel='stylesheet' href='css/demo-css.css' type='text/css' />\n";
321     outputMessages += "</head>\n";
```

Let us see how the header is assembled. The idea is fairly simple. It starts with a variable output messages, which starts of as null or empty, then the entire webpage that we want to produce the markups are actually included in the string. And using the plus-equals operator, they are concatenated to the previous value of output messages. So, initially it was null, and then one-by-one the string gets added to generate the full string. Note that we add \n at end of the lines.

Because of this backslash string `\n`, the new line character, our HTML page appears as if handwritten. But when this page that we have apparently handwritten goes to the browser, these new lines have no meaning. The browser will only respond to `
` tags. Also note that because our HTML itself requires double quotes, these double quotes must be escaped. To avoid misinterpretation of these double quotes.

If you observe, what we are really doing is simply writing the entire webpage ourselves but within Java and the functions that we write are going to assemble the header part and just start up the body part. We have introduced this 'div' to divide our HTML into sections and the sections will be named using the class attribute. In this case, this section is named as welcome block. By welcome block we mean the heading and brief information that is displayed that makes our user comfortable.

It informs the user of what exactly is this application about. After you mark the start of the welcome block, you have the h1 title as usual, you have some message for welcoming the user and giving information about it. And yes, the br tag, very necessary, and enclosing forward slash div tag. This forward slash div and this div, together mark the region of the HTML that we want to treat as a block called welcome block.

Once all of the output has been collected into output messages at this point, we simply print it out and then we print out a HTML header and comment. What we are really doing? When we say that the Java program creates an HTML page is simply writing down the page using Java constructs. That is all. If you look at the dual registration HTML part of the story again it is the same thing.

You start with the output messages as empty and go on assembling the registration information into this string variable. By assembling we mean; the new information is concatenated to the previous information, which is why we have a plus-equals operation here. Again, we are using the div tag with a class name as reg block to demarcate the region of HTML that we want to treat as a registrations block.

We will use this treatment part of the HTML as a registration block or welcome blog or whatever blog, in the corresponding stylesheet. As can be seen, the entire code is simply about writing the HTML page using Java constructs for string manipulation. It is this that gives us a web page. As a result of when the server responds with a web page, this is how the web page gets generated. We had already seen this before. **(Video Ends: 1:00:14)** That brings us to the end of this session. Thank you. See you in the next session.