

**Internet Technology**  
**Prof. Indranil Sengupta**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**  
**Lecture No #16**  
**Extensible Markup Language (XML)**

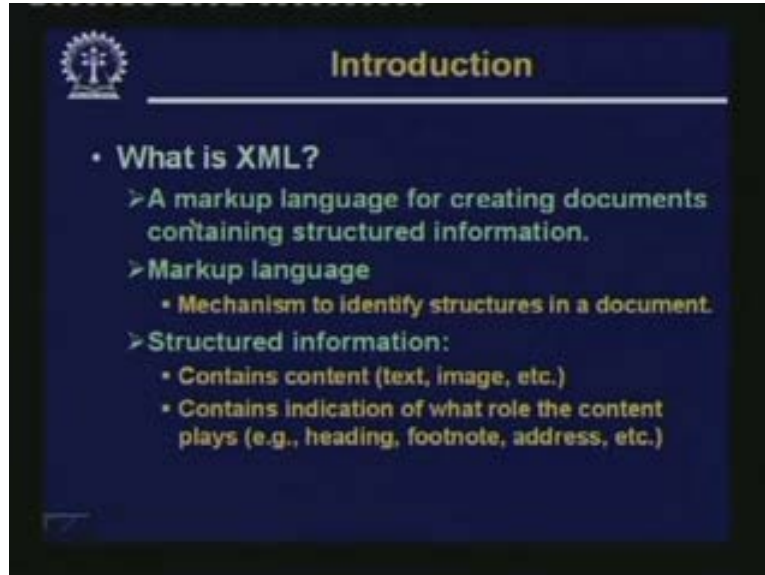
Welcome back. We have been talking about the issues of design of web pages. We had talked about html in the last few lectures. And we had also talked about that there are some other technologies which have come up subsequently. This html is not the only way in which you can go ahead in time. There are other more powerful and more flexible solutions in language which has come up. In today's lecture we shall be taking up one such language for the purpose of discussion.

(Refer Slide Time: 01:30)



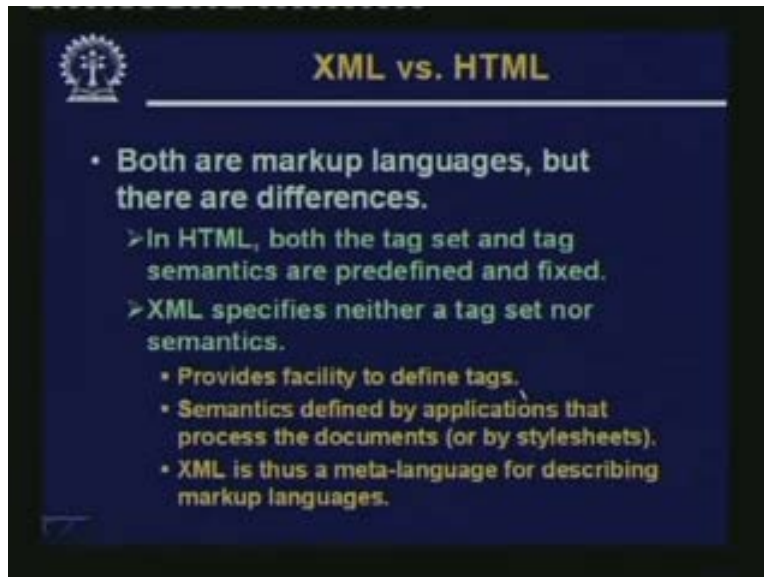
And this is extension extensible markup language which is more commonly known by its acronym xml.

(Refer Slide Time: 01:38)



So first let us try to understand what is XML? See we had seen html earlier. Html is a language using which you can create the web pages. It is a markup language. Markup language means, we specify the textual content of the document. As well as we specify how the textual content should be structured or should be viewed on the screen. The concept of the tags and attributes come from there. So this is what a markup language is. Now XML like html is also a markup language. So here lies a similarity. Let us see. So XML is also a markup language. Well although here we are talking about creating documents. We shall see later that XML has been used not only for creating document but for many other applications as well. There is some structured information that we can capture through tags like html. Markup language is that, markup language use of tags is a mechanism to identify structure. For example a begin paragraph and end paragraph tag will tell you that, well this is a paragraph in the document. Begin h1 and end h1 will tell you that this is a heading in the document. So these are structure information which you can extract. So structure information will contain content which can be text, which can be other form of information also, image. And also it should contain of what role the content plays. For example I gave you some examples, heading, paragraph, address. But one problem with html is that the set of tag is limited by the definition of html. If for example I want to define a new tag for country, for city, I do not have it. But in XML we can do it. We can define our own tag we shall see it later.

(Refer Slide Time: 03:57)



So if you compare now, XML and html first thing is that both are markup languages, there are differences as I had mentioned. This html is a fixed language the tag set and the meanings of the tags are predefined. They are fixed and you cannot change it. The browsers have been designed following these rules. Even you cannot change it in the browser you cannot tell the browser that h2 tag you should display it in italics in color red that we cannot tell the browser. You will have to modify the document itself in order to make this into effect but in contrast XML specifies neither a tag set nor semantics.

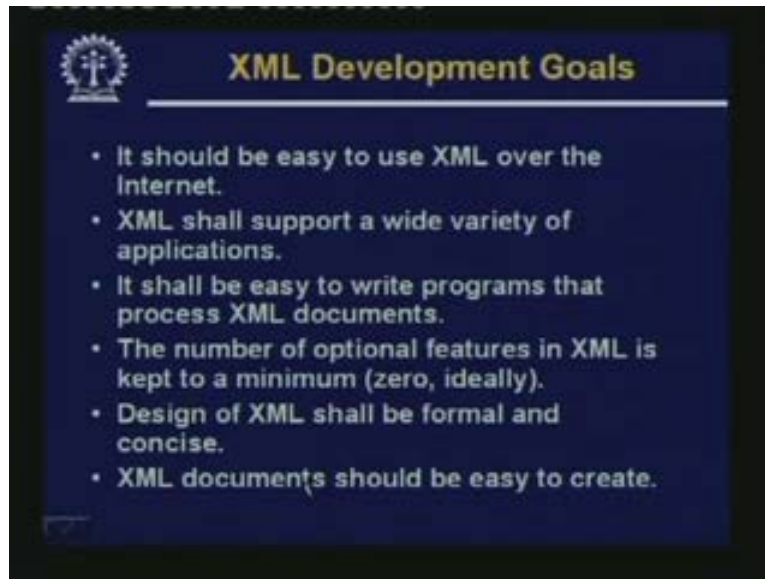
XML is a totally open language and totally flexible language it does not specify anything by default, depending on your application domain your context you will have to define everything and it is entirely up to you can customize XML as you want it to be. So XML provides you the facility to define tags. It can allow you to define semantics which can be defined by applications for example this html means I repeat. Html is meant for creating documents XML is meant for creating some content. This content is not only meant for viewing on the browser.

There are a much wider range of applications of xml, as you can see some typical application later. The idea is like this. Whatever you define in XML you can write a program later may be in C, C++, Java, whatever that program can read that document and can process it. So the application there will be of course some application programming interfaces to extract the structure information from the XML specification. And the semantics can be defined by the application that what a particular tag means that the application can define or can understand.

So XML is therefore more like a meta language it is not a fixed language which defines everything it is a language using which you can describe markup language. For example using html you can define html because html has a fixed set of tags you can define all of

them in htm, in that means you can define al of them in html and whatever you get is an html implementation using xml. Similarly using XML you can do a host of other things host lot of other things.

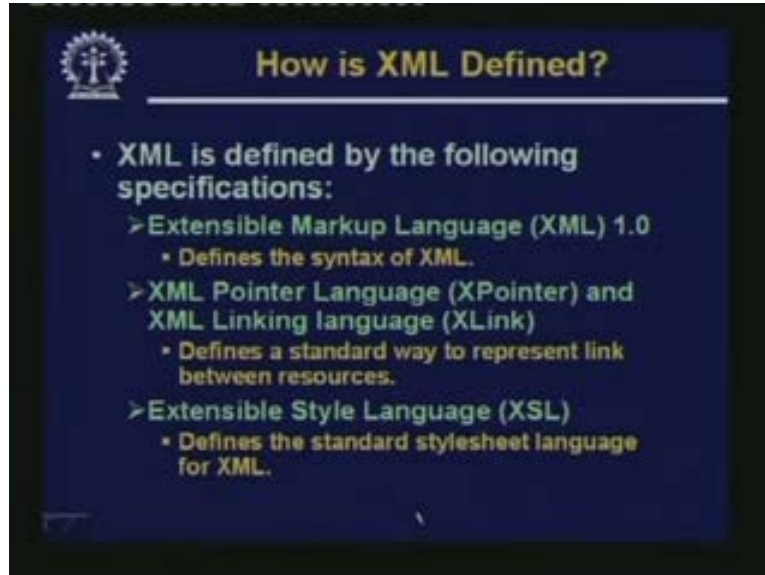
(Refer Slide Time: 07:10)



But the primary goals of XML if you look at it, you will understand a few things. The first was that initially it was meant to be used over the internet. So the first goal was it should be easy to use over the internet. This also means that the browsers you see around us they should understand xml. There should be some complaints in deed most of the modern browsers can understand xml. XML shall support a wide variety of application, this is a secondary objective. This application does not necessarily mean that only viewing the documents on the browser. It shall be easy to write programs that process XML documents. These are these applications which are running using XML specifications. The number of optional features in XML is kept to a minimum ideally it is zero.

See optional feature means if you look at the at the html specification, you will find the some of the specifications were referred to has been mandatory. But some were set to be optional and there was a problem with the optional specifications. Some browsers implemented some optional features, some other browsers implemented some other set of optional features. So if you create a document using the optional features there is no guarantee that can interpret it in a correct way. So if there were no optional features at all there would be no ambiguity. So XML does or aims to just that. Design of XML shall be formal and concise it is so and it should be easy to create. There are many XML document editors available now a days using which you can create XML documents.

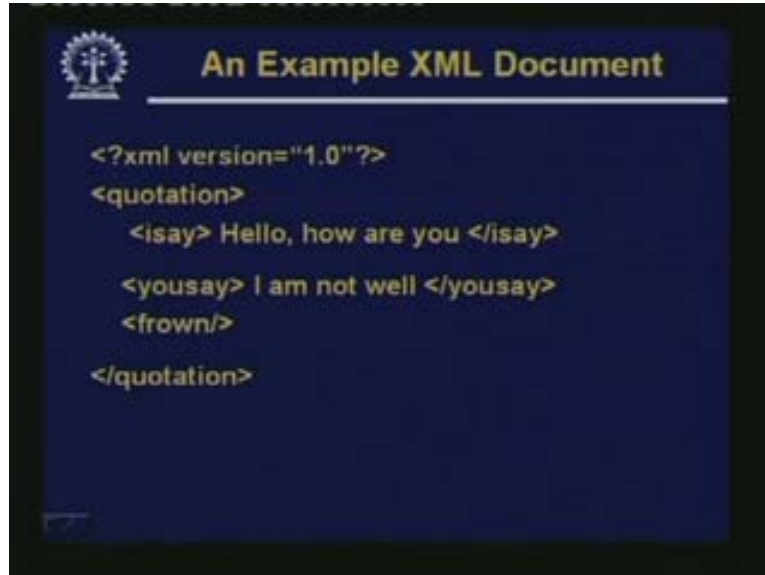
(Refer Slide Time: 09:13)



So let us see how XML is defined. XML is defined broadly using three different specifications or specification languages whatever you call. The first is the basic XML specification. This is the extensible markup language version “1.0”. This defines the syntax of xml. The language constructs how we have to how you need to write the constructs of the language the syntax. The second thing is there is something called XML pointer language in short x pointer; XML linking language in short x link. These are still undergoing some refinements. Now this xpointer and xlink, these are some specifications or languages using which you can create some thing similar to hyperlinks as you create in html. The standard XML specification does not support hyperlink.

You have to use xlink specification for that. So this xlink or xpointer this defines standard ways to represent link between resources and the third one is extensible style language or xsl. This is something similar to the html style sheet that we were discussing in the last class. Here you can define the standard style sheets and this language specifies how means the exact format and syntax using which you can do that. The advantage of having xsl is obvious. Suppose your main application is to use the browser for viewing information. Then you will be creating a style sheet xsl specifically for browser related definitions and syntax semantics whatever you call. So how to compose paragraphs headings center different kinds of alignments, fonts, etcetera.

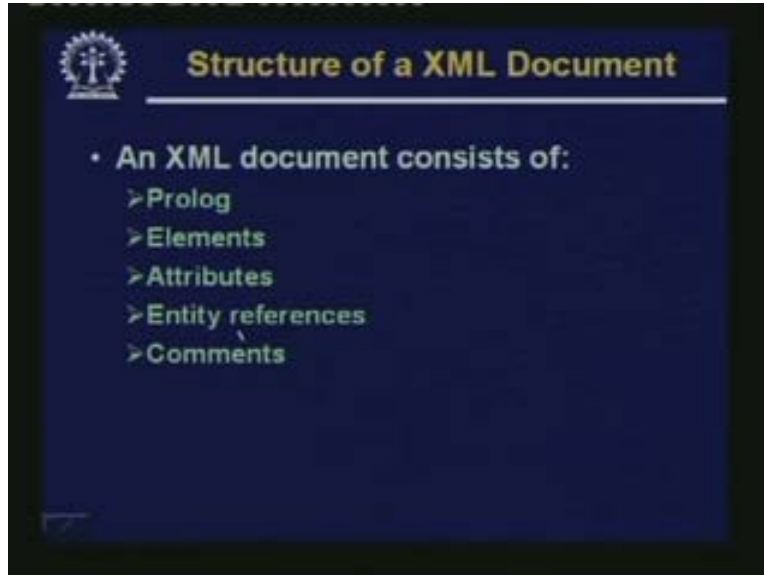
(Refer Slide Time: 11:30)



A very small example an XML document looks like this. It starts with a beginning line. This is sometimes called as an XML prolog. This tells you that this is an XML document it begins with less than question mark followed by the key word xml. Version specifies which version of XML it is followed by end and inside the XML there can be several tags just like html like quotation, begin quotation, end quotation, Isay, end isay, yousay, end yousay. See these tags look very peculiar it is indeed peculiar. They were not part of the definition I have defined it according to my need. This particular example I have taken may be it is a portion on except from a dialogue that is going on between two persons myself and you. See whenever I am enclosing some text between the tag isay. This is means I am telling you something between yousay means you are telling something. There can be a large number of alternates of isay and yousay in the total document that will constitute the whole conversation.

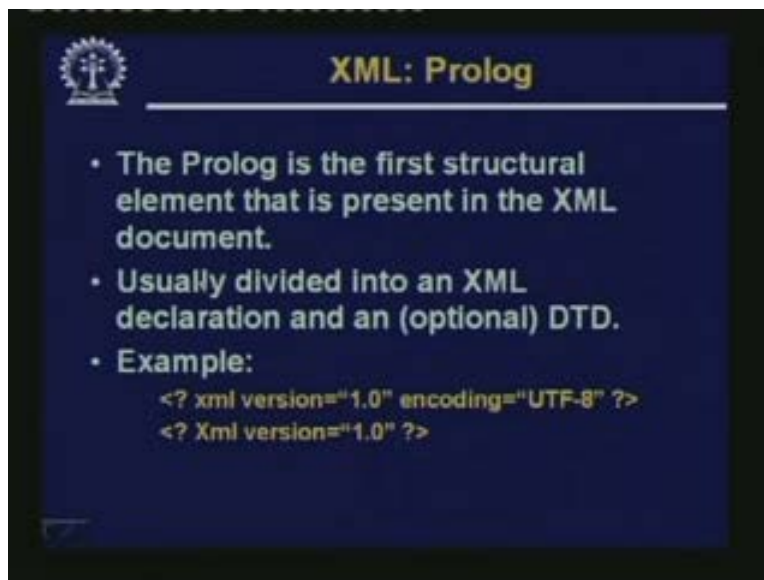
But I am calling quotation, you can call it something else also. In xml in some cases you may have a pair of tags with nothing in between it is called an empty tag. There is a way to specify an empty tag it is like this the name of the tag followed by slash. This is of course equivalent to begin frown end frown. So instead of writing this frown twice. I can simply write frown followed by slash. This will mean that it is the frown tag but it is presently empty nothing is there in between. So this gives an idea about what or how means html document will look like there are some tags. Well I had used some tags according to my need and I can understand what the meaning of this tags are but may be you will not understand. So the tags will be defined depending on the application which will be processing it or the person who will be viewing it or looking at it. This is the basic idea.

(Refer Slide Time: 14:16)



So talking about the structure of a XML document there are 5 different components of the document. Prologs, elements, attribute, entity references and comments. Let us see what these are.

(Refer Slide Time: 14:40)

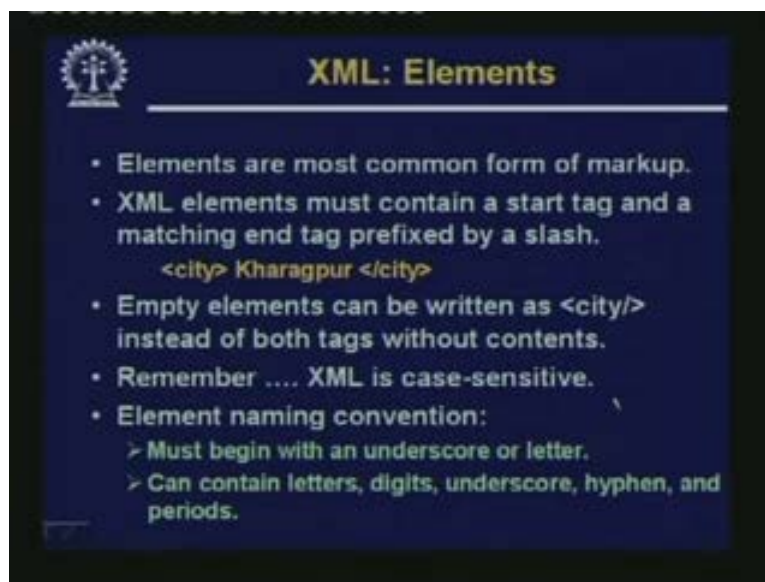


First is prolog. Prolog as I said with the help of that example. That is the beginning you need to specify that this an XML document that is the prolog. It is the beginning or starting of the document that tells you the prolog has to be the first structural element that is present. First structural element means before that you can have comments. Comments are just like html, so there is no confusion. There you can have blank lines. But the first

structural element will be the prolog. Prolog is sometimes divided into an XML declaration and an optional so called data type declaration that we shall see later.

Some typical examples of prolog definition are this. The second version we have already seen earlier the first version has an optional tag. This is the DTD I have said. This specifies some encoding rules. This is an XML file version "1.0" it uses some encoding rules UTF8. UTF8 is a standard encoding rule where some special character can be encoded by some special escape sequences and the like. So that when you are processing it, just looking at the prolog you will understand that what kind of special characters and their encodings you are likely to encounter in the document.

(Refer Slide Time: 16:13)



These XML elements are like tags. So XML elements they are the most important form of markup or the way of specifying some sections of the document or text whatever you are correct, you are doing or writing. Now XML elements when you are using, they must contain a start tag and a matching end tag just like html. You can say that in html the elements were predefined and you are calling them tags so begin tag end tag there are ways of doing it. End tag starts with a slash with the same name. So here also that syntax remains the same. A small example an element we are calling it city end element slash city in between some text.

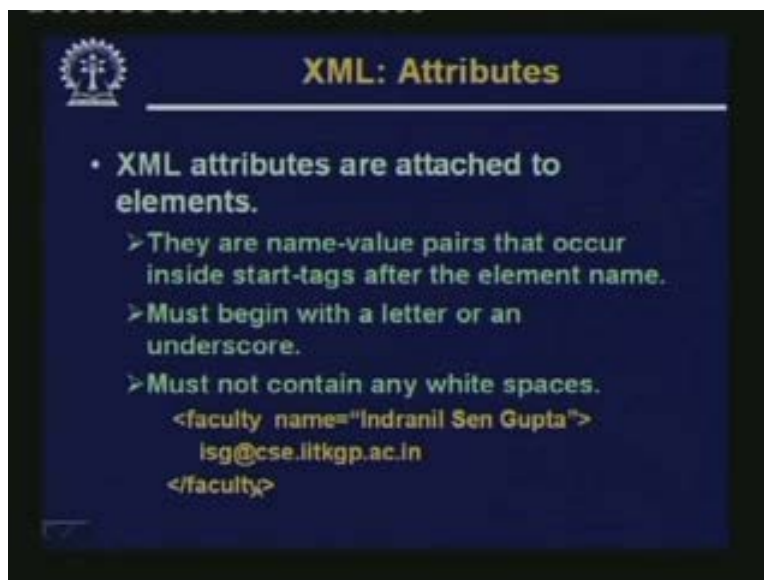
Well this text can be arbitrary, but in the context of XML we understand that this Kharagpur is a name of a city. Some application which is reading the data will understand that this Kharagpur is part of an element called city. So the application understands that this is a name of a city. So semantic is also captured here very nicely. And as I mentioned in that example, that if you have empty elements you can write them by putting a slash after the name of the element. Well here alternatively you can also use both the tags without any content in between blank and one thing to note is that unlike html XML is



case sensitive. So when you are defining an element city lower case c i t y and capital C I T Y are not the same.

You must follow the same case convention when you are using xml. When you are just using XML for some other applications. There is a naming convention for the element. That means when you define for example I have given the name city. So how you can define names? There are some simple rules. Rule says, that the name must begin with an underscore or a letter. Letter means a correct a to z. But within the name you can have any number of letters, digits, underscore, hyphen and dots or periods. So any combination of this will constitute the name of the element. So it is an element you can construct using any combination of these.

(Refer Slide Time: 19:09)

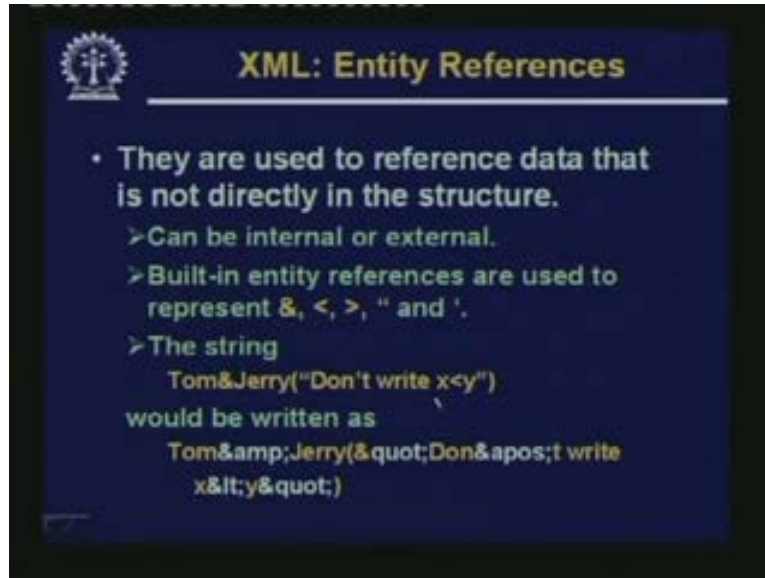


There is something called attributes, this attributes are also quite similar to what is there in html. In html we had tags; along with tags we had attributes. Whenever we have the begin tag along with the begin tag we specify the attributes. Here also we do something very similar the syntax is very similar here also. Here the tags are called elements. So we say that attributes are associated with elements. So XML attributes are attached to elements they are mostly name and value pairs that occur inside start tags like html after the element name. What I mean is that something like this. Suppose in this example we have defined an element whose name is faculty.

And in this start faculty tag element you can say I have defined an attribute. The name of the attribute is n a m e name and indranil sen gupta is the value of that attribute. And within this I can have anything. For example here the body contains the email address but it can contain anything. So name value pairs will constitute attributes. So this simple example shows that we have associated one attribute with this element faculty. Must begin with a letter or underscore. The attribute name must not begin with anything else.

This is the constraint and must not contain any white spaces. You cannot give any blank in between. This equal to before and after should not be any blank.

(Refer Slide Time: 21:12)



Well entity references, see entity references actually mean that when you are creating a document you define some tags. Tags normally use less than greater than for defining tags. So use some of the symbols that are available in your alphabet that are available in the key board to define something special. Now if in the document those same symbols are appearing you normally use some kind of escape sequence to indicate that well this is that particular character not the starting of a tag less than for example. So the entity references as it is mentioned under XML actually talks about exactly that. They are used to reference data that are not directly in the struct.

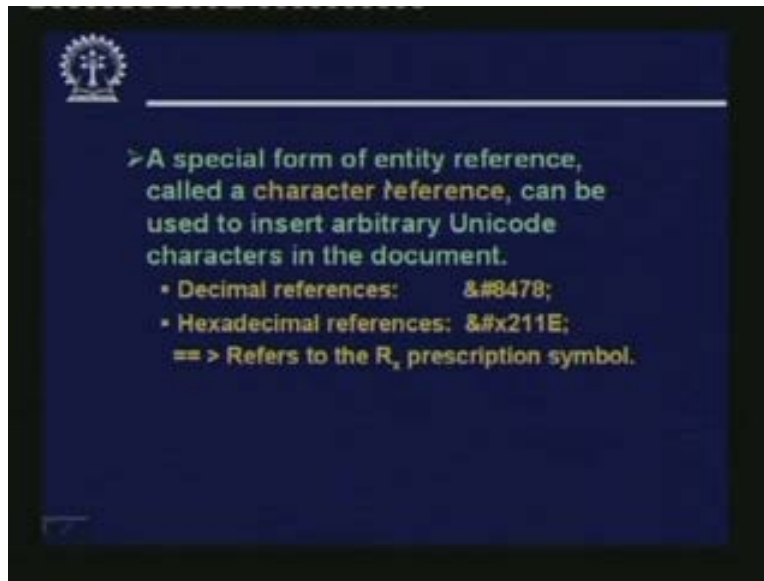
Not directly in the structure means you cannot normally use those data or those symbols in the structure because they mean something else. But in order to use that you will have to do something else. So there are two things. This can be internal, this can be external. There are two ways of doing it. So internal means the entity references whatever you do; that will define as part of the same file and external as the name implies this will be in some other fact. But in addition to internal external there are other ways of mentioning. We will see it through examples later. Built in entity reference something internal external there is something builtin. This is already there.

Builtin means these are some special symbols which we use in XML ampersand, less than, greater than, quotation, double quote and apostrophe, single quote. So a small example follows. Suppose I have a string like this Tom and Jerry bracket quotation do not. There is an apostrophe in between. Write x less than y quote bracket closed. Now if you write a string like this, this ampersand double quote single quote less than, this will totally confuse the XML parser or processor because they some other meaning to xml. So

you will have to escape out this using special escape sequences and specify it in a slightly different way. The same text you will be writing like this.

Instead of this ampersand symbol you will be writing ampersand amp semi colon. This ampersand is the escape sequence here all special symbols will start with ampersand then some name small short name followed by a semicolon. So ampersand amp semi colon is this jerry then this quotation comes double quote. This is ampersand quote ampersand semicolon. Don, d o n then the single quote comes this is ampersand apostrophe semi colon. Write x less than y less than is again It double quote at the end this is quote. So in this way where ever there is a special symbol that you replace by an escape sequence which starts with ampersand followed by some short name for that particular symbol followed by a semi colon. This is the XML builtin entity reference.

(Refer Slide Time: 25:05)

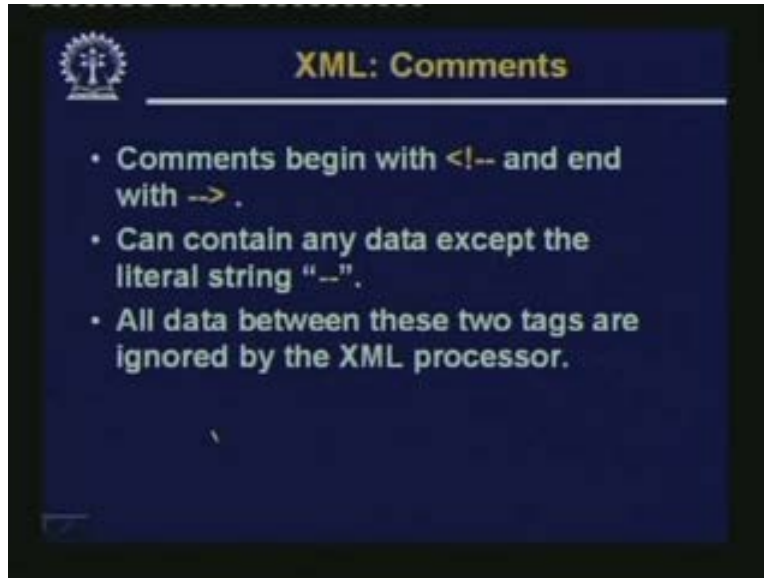


Now there is a special form of entity reference also which you can use. This is called a so called character reference. Now using character reference, you can insert arbitrary characters in the documents. See you know this Unicode is a 16 bit character code which can be used to virtually represent any character in any language. So this allows you to insert arbitrary Unicode characters in the document. Well I am giving a small example. How to do that? You can specify the character code in either decimal or in hexadecimal. In decimal you do it in this way. It as usual starts with ampersand.

This hash symbol indicates that this is a character reference followed by 8478 means the corresponding decimal equivalent of the Unicode 16 bit number; followed by semicolon. Hexadecimal is similar, only you use a x before the hexadecimal number. Ampersand hash x 211 E. These two actually means the same number and this refers to the prescription symbol which the doctors give on the prescription that r a small x. That r x is a special symbol that has Unicode number 8478 in decimal. So this is just an example

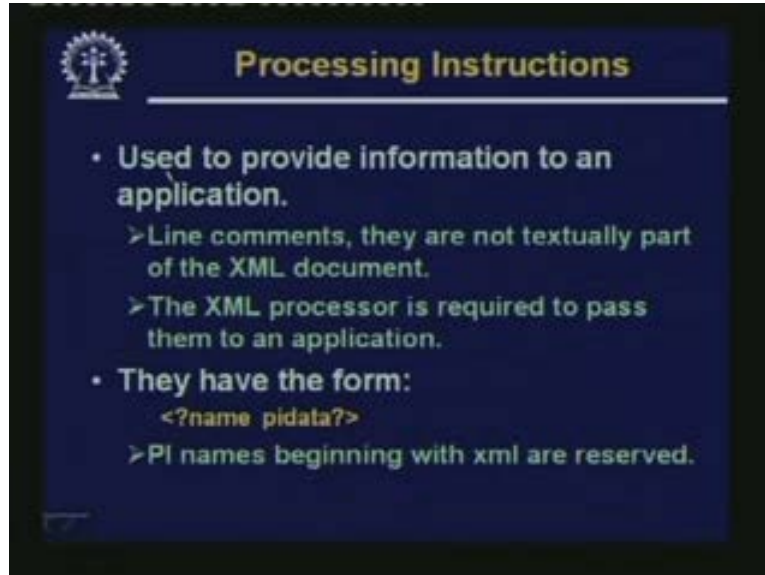
which shows that if you give this r this means 8478, actually this prescription symbol will appear on the browser if you are viewing it on the browser screen.

(Refer Slide Time: 26:48)



Then comes the comments. Comments can be any text which starts with this particular pattern less than exclamation sign and followed by two minus signs dash and ending with dash dash slash. The only restriction is that you can contain any text in between excepting the two consecutive dash characters. So all the data that is present between these two tags will be ignored by the XML processor, they will be treated as comments. These are only for human readability. Readability and the understandability of the document you have created. Nothing to do with the XML processor. When the XML processor parses the file and sends you can say the values to the application along with the tags. These comments are totally ignored.

(Refer Slide Time: 27:50)



There is something called processing instructions. When some values are sent this processing instruction can be used to provide some additional information to an application. See this additional information can be of a number of different types. For example I am giving a very simple example. Suppose you have an image for some image you may want the application should open it just using bmp editor. Sometimes you may want that the application should open it through a jpeg viewer it is available. So that value should be sent as a parameter. It is possible to sent it, but what ever data you are actually sending is an image.

But how the image you want the application to handle that additional information also you can add or you can incorporate with the definition. So this provides information to the application like comments they are also not textually part of the XML documents. XML document does not have bearing with this personal instruction. These are important only for the application which will be processing the html documents. So the XML processor will simply forward or pass them to the application that will be processing these processing instruction. But otherwise the XML processor has nothing to do with them directly.

They have syntax like this less than question mark they are special XML commands or tags you can say. Here you can give the name followed by then. For example, for the image as I told you; you can give the image type, you can give bmp or image type jpeg. This you can give as a additional information to the application. So you can have as many as this kind of processing instructions declaration as you need and you can have them to make the application behave in a way you want. There are number of instructions that start with xml. However they are reserved, this as a user you are not allowed to use. They are reserved for particular use for certain particular applications.

(Refer Slide Time: 30:46)

**CDATA Sections**

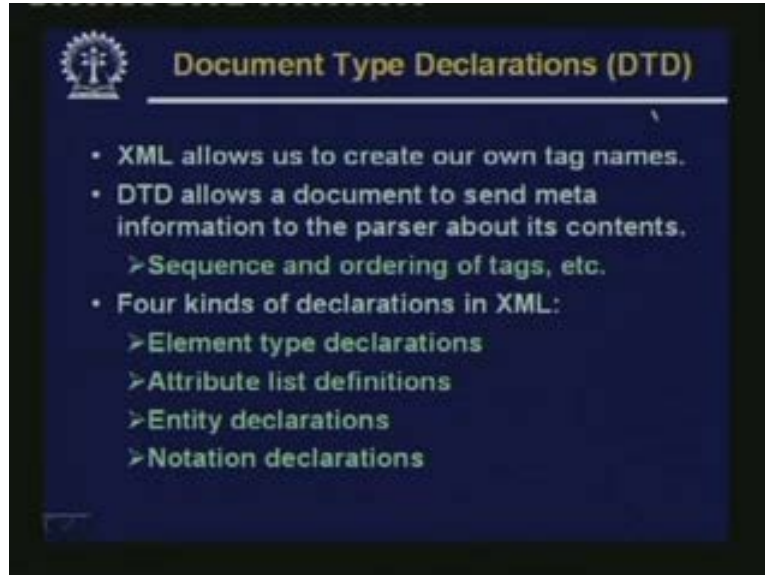
- A CDATA section instructs the XML parser to ignore most markup characters.
- An example:

```
<![CDATA[
temp = *p;
*p = *q;
*q = temp;
if (temp < 0) temp = -temp;
]]>
```
- All character data in between is passed to the application without interpretation.

CDATA section is sometimes used in XML to bracket out a portion which you do not want to. You can say use escape sequencing like the CDATA section in xml. It instructs the XML processor to ignore the markup characters that are appearing there. Because otherwise those markup characters would be replaced by the escape sequences. A very small example is given here. Here I have a small segment of c code which contain some special symbols like asterisks like less than which may have some other meaning too. XML this asterisk normally do not have meaning. This less than has, so if you put it in the CDATA section, this is how we put in less than exclamation square bracket CDATA. Again another square bracket begin at the end double square bracket ending and a greater than.

So if you give it in this way whatever is enclosed all special symbols like less than will be retained as it is. They will not be replaced by the corresponding escape sequences. So particularly when you are composing some program text or something to you really do not want them to be replaced by something else. You try to return them. So CDATA section can be useful in those cases. So all character data is passed to the application without any interpretation. That means you do not try to interpret them. For example within this CDATA section there can be some ampersand symbol also. Normally what the XML parser will do? Whenever the XML parser sees an ampersand symbol, XML parser will try to see what follows that. That is a special character definition and it will try to replace it by something else. But here nothing is replaced. Verbatim whatever is present is been sent to the application within the CDATA section.

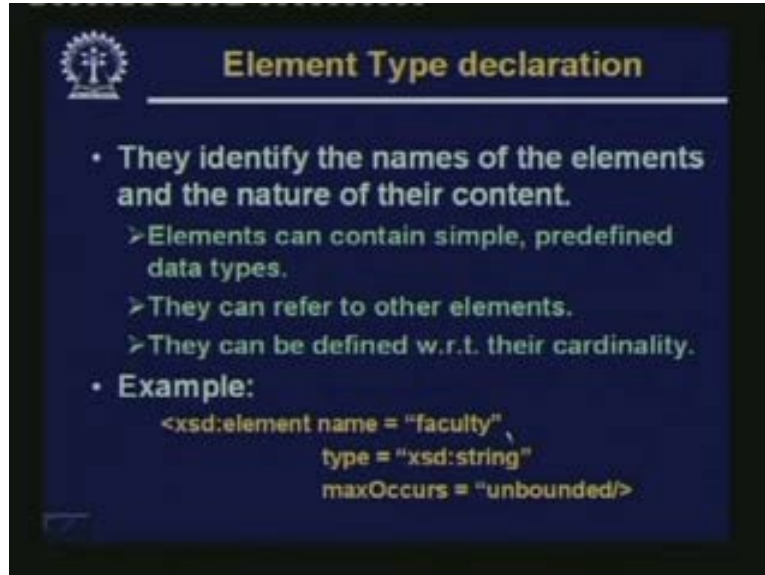
(Refer Slide Time: 33:01)



Next comes document type declarations or DTD. Say XML as though as though it may seem that through element declarations it allows us to create our own tag names. Document type declarations allows a document to send Meta information to the parser about it contents. Meta information means that you send some semantic information to the parser that well these are the tags and this is the meaning of the tag. This how you should interpret the tag this kind of Meta information can also be sent. Like I am giving a simple example, you think of a browser. Browser is a one very classic example of an XML application an application that is using XML specification. A web page written in XML that is been downloaded the browser tries to display it browser is now an application.

So browser is using an XML parser to extract some important information from the XML specification and try to display them in a proper way. Now DTD document type declarations are very useful without the document type declaration the browser will not know that. How to display this particular item on the screen bold centered underlined or how DTDs are important due to this. It also tells you that if there is any important sequencing and ordering of the tags. This also may be mentioned that tags must appear in this particular sequence. You cannot have any arbitrary usage of tags. Now broadly four kinds of declarations are there in XML right you can define elements, you can define attributes as a set of elements, you define something called entities and notations. These are the four different kinds of definitions that we encounter in XML specifications.

(Refer Slide Time: 35:17)



The slide features a dark blue background with a white logo in the top left corner. The title 'Element Type declaration' is centered at the top in a yellow font. Below the title, there are two main bullet points in white. The first bullet point is followed by three sub-bullets, each preceded by a yellow arrow. The second bullet point is followed by an XML code snippet in white.

- They identify the names of the elements and the nature of their content.
  - Elements can contain simple, predefined data types.
  - They can refer to other elements.
  - They can be defined w.r.t. their cardinality.
- Example:

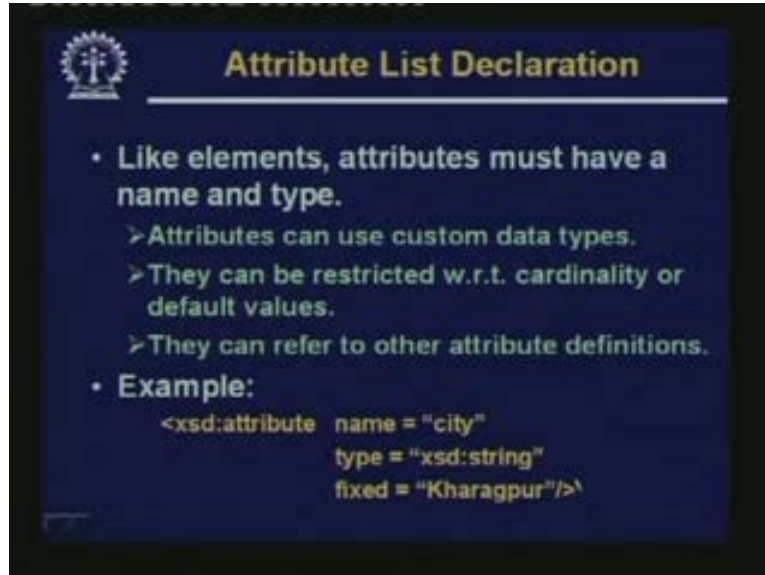
```
<xsd:element name = "faculty" \
  type = "xsd:string"
  maxOccurs = "unbounded"/>
```

Let us see first is element type declaration. They identify the names of the elements and the nature of the content earlier we had seen what an element is and how you can use an element in a XML document. Let that city Kharagpur end city that was one example. Now here you see how you can define an element and XML document. Elements identified in names of the elements and the nature of the content. So I told you in XML there is no predefined element or tag everything I need. I will have to define myself. So this is a way of defining, so elements can contain simple predefined data types they can refer to other elements. They can also be defined with respect to their cardinality. Cardinality means how many times the element appears in the XML document.

Because I may say that a particular document city should not appear in the document more than twenty times or it is unbounded. It can appear any number of times. These kinds of specifications I can also mention through a small example. This defines an element whose name is faculty. Xsd colon element is the keyword used to define elements. Name equal to the name of the element you are trying to define, then type there are several predefined types under xsd again. Xsd colon string means, it is a string. Max occurs is another attribute which says that it is an unbounded that means that there are no limit to the number of times. The faculty element can appear in my XML document. So this way you can define the elements.



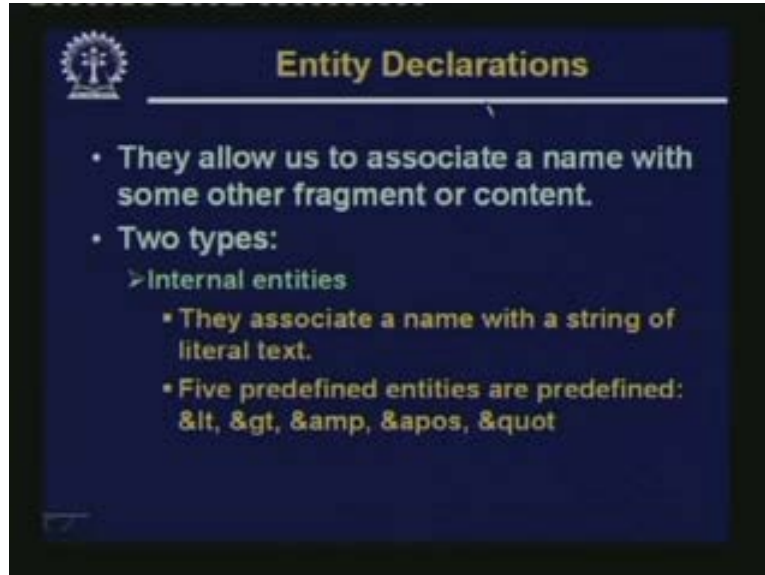
(Refer Slide Time: 37:26)



The slide features a dark blue background with a white logo in the top left corner. The title 'Attribute List Declaration' is centered at the top in a yellow font. Below the title, there are three bullet points in white text. The first bullet point is 'Like elements, attributes must have a name and type.' followed by three sub-points: '>Attributes can use custom data types.', '>They can be restricted w.r.t. cardinality or default values.', and '>They can refer to other attribute definitions.'. The second main bullet point is 'Example:' followed by an XML code snippet: '<xsd:attribute name = "city" type = "xsd:string" fixed = "Kharagpur"/>\'

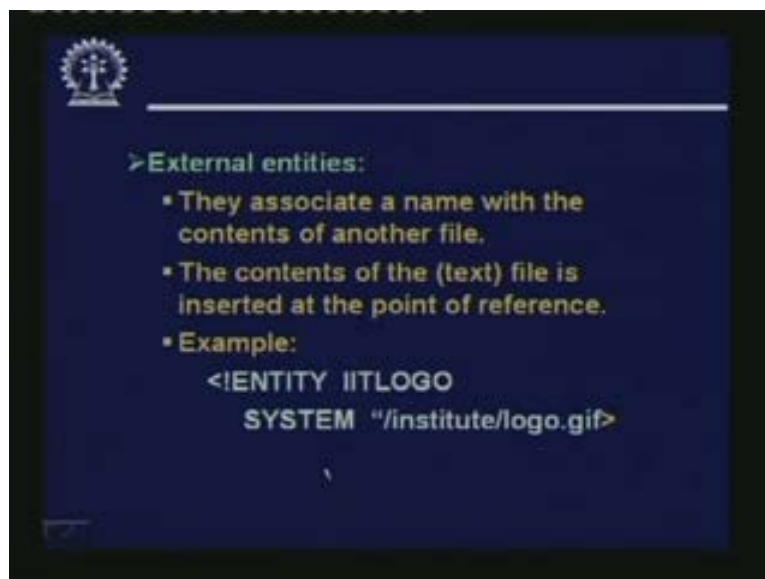
Next come attributes. How to define attributes associated with the elements. Like elements attributes must have a name and type. Attributes can use custom data type also they can also be restricted with respect to cardinality. As I had mentioned before and also some default values you can specify they can also refer to other attribute definitions. Like one small example follows here. Xsd attribute this keyword tells you that you are trying to define an attribute the name of the attribute is city. Type is again string fixed Kharagpur means this is a default value and these are restricted value cannot change it. But if you do not give this then anyone can use any kind of city by default. For example you are you are creating an XML document that is keeping track of all the bank branches in Kharagpur. So for them the city will always be Kharagpur you can define them as fixed type, fixed equal to Kharagpur for them this kind of declarations are relevant.

(Refer Slide Time: 38:41)



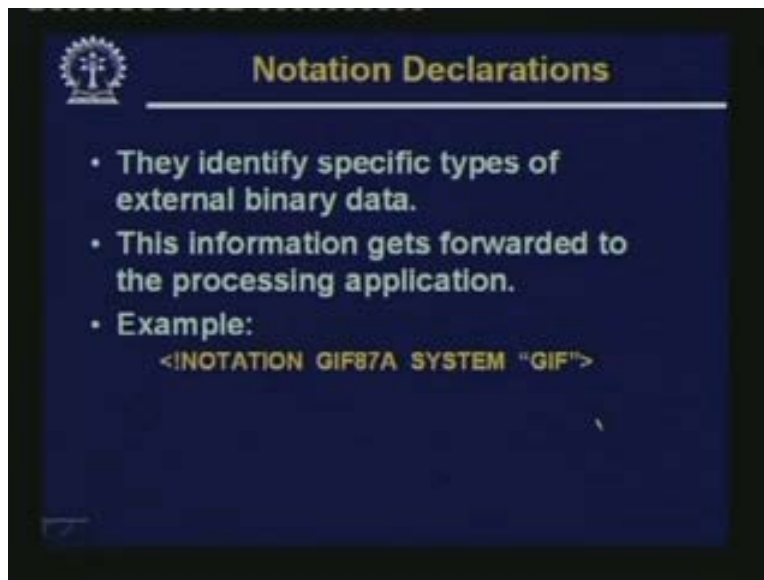
Entity. Entity will allow us to associate a name with some other fragment or content. Like we have already seen internal entities. Like you are replacing some special characters by their escape sequences. This is one example of entity. That means it is like a macro substitution you have something to replace it with to something else. And later on when you again need it back you replace it back. It is like a macro substitution and substitution. This is sometimes called an entity with which some string or something else is associated you replace the entity by that string. Essentially it means the same thing, it is like a macro substitution. You have a symbol or a string you are replacing it with by some other string. These are internal entities which you can define as part of the xml.

(Refer Slide Time: 39:51)



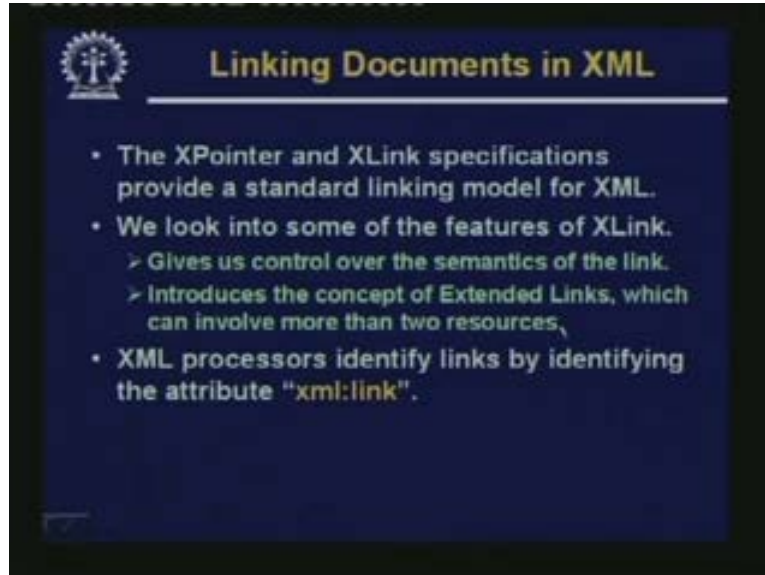
External entities are those which associates a name with the contents of another file. This is also like a macro. But you replace it with something which is not there in your XML file that has to be brought from somewhere else. So that somewhere else, once you fetch it the contents of that will be inserted at the point of reference. There is one simple example I am giving. Here I am defining an entity. Entity IIT logo, this is the name of the entity I am giving. I am saying system is a keyword followed by a link which tells you where this logo is found this is the URL. This is actually a URL institute slash logo.gif. So now in my XML wherever I use this IIT logo entity that will get replaced by that gif file gif image. These are very convenient way of replacement and this external entity will allow you to do it from other file not from the same document.

(Refer Slide Time: 41:10)



Notation declaration is another thing which sometimes simplifies the task of the application. So using notations you can identify some specific types of external binary data. Inside one example we will illustrate. Take this example. This is how we define a notation definition. This is the name gif 87, this is an image format you know system. This very complex image format 87, this is the complex stream. But you tell the application that it is nothing but a gif image, you can open it using any gif image viewer program. So the name may be any complex thing. But you give some kind of additional information to the application using this notation at well this you can denote by this this is a shorter form of this. This is kind of short shorter form notation is the most typical use of notation declarations.

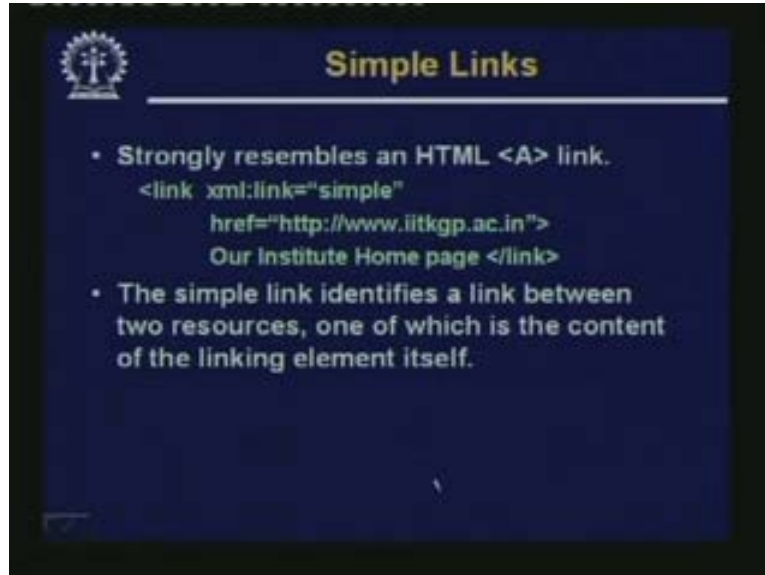
(Refer Slide Time: 42:29)



Now let us see how we can document some means you have created a document. How we can link a document to other documents? Just like we can use hyperlinks in html. We have a document we can define an anchor. We can provide a link to some other portion of the document or to some other document. So in XML how do you do that? XML in itself does not have this feature. So you need either the xpointer or the xlink specifications which I had mentioned earlier. These are some kind of standard linking model for xml. So we specifically look at some of the features of xlink. Now well using xlink you can define hyperlink. But it is something more than the kind of hyperlink you use in html. It also gives you control semantics of the link. Semantics means the meaning means, in html semantics is only one you click something comes to you that is the only meaning.

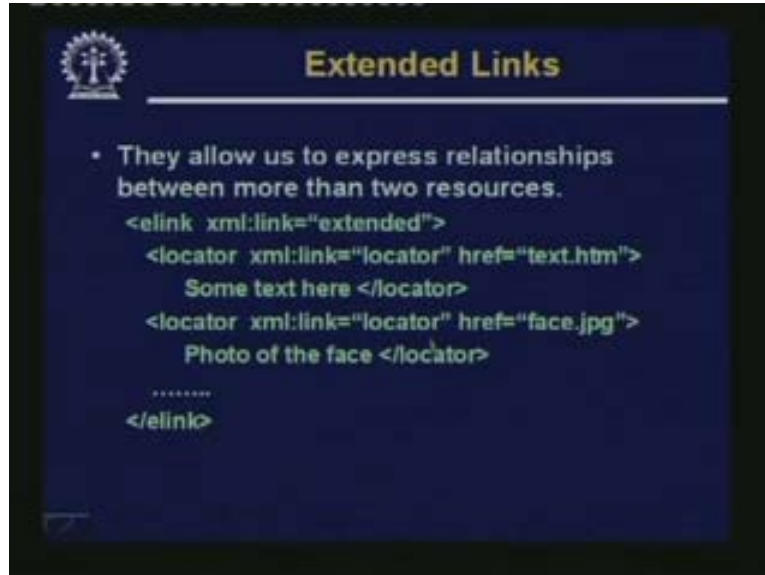
Your URL which you are clicking the hyper link that corresponds to some other document which is located on the web server. That is it. But here it can be something more. It may not be something to do with you have to bring something from the webserver and view it on the screen. Because it may not be internet application at all. It can be something more general. So let us see, so for this purpose they have introduced something called extended links. We will give an example for this. Extended links can involve more than two resources. The conventional link or hyperlink involves two resources from one you provide a link to the other. But extended link can involve several resources together. XML processors can identify these xlinks declarations by this special keyword XML colon link. Let us see how we can do this?

(Refer Slide Time: 44:50)



First let us look at simple link. Simple link is very similar to the html anchor tag which provides normal hyperlinks you start with a link. Link is a there is an XML specification link is, it is an element. XML link is the name which you specify a simple. This is a simple link which is followed by the text or the URL which is clicked then the actual clickable text. So this is the URL followed by our institute homepage is the clickable text. Now this simple link actually identifies a link between the present resource and the resource whose URL is given here [www.iitkgp.ac.in](http://www.iitkgp.ac.in). So this is very much similar to the html link mechanism. But when you look at extended links you will see that it is something more. So this XML is not something to do only with viewing document on the browser and clicking them to get something else.

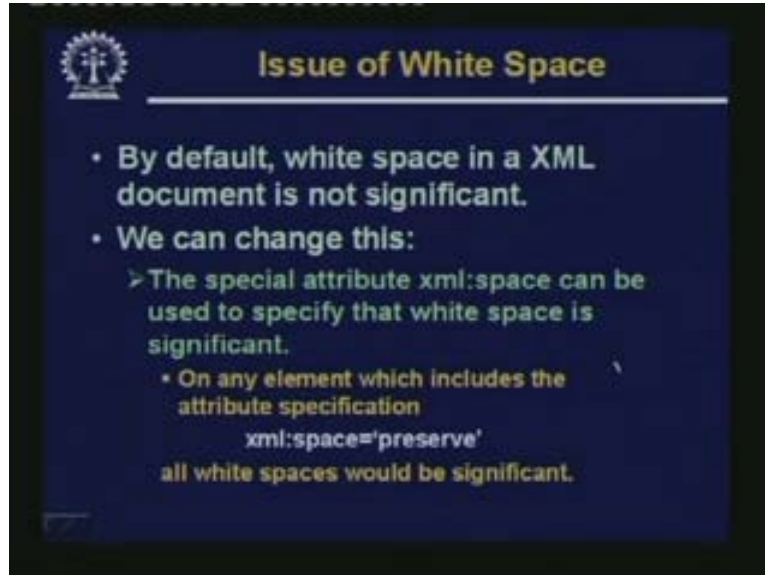
(Refer Slide Time: 46:18)



The slide, titled "Extended Links", features a logo in the top left corner. It contains a bulleted point stating: "They allow us to express relationships between more than two resources." Below this, it displays XML code for an extended link. The code starts with an opening `<elink xml:link="extended">` tag. Inside this tag, there are two `<locator xml:link="locator" href="text.htm">` tags, each followed by text: "Some text here" and "Photo of the face". The code ends with a closing `</elink>` tag. There are some asterisks between the two locator tags in the original image.

So extended links as I had mentioned they allow us to express relationships between more than two resources. So for this you use the elink tag and you specify the attribute XML link as extended. And we use another tag inside elink called the locator. You see we are using two locators well two I have shown it can be more also it can be more than two for a single elink. Now this, within this locator you specify XML link equal to locator. Locator is a keyword. Some document you specify and also some text well it depends on the interpretation. This is not for the conventional browsers. Some browser may interpret it as follows. If that browser is your target application, that in two different frames you will be displaying the two links together simultaneously. You click one link, you go to two different places together and the two different things are getting displayed side by side on two different windows. So these kinds of things are possible if we go for extended links, so it is a much more generalization of the conventional link where by a document is getting linked to some other document through a single link.

(Refer Slide Time: 47:49)

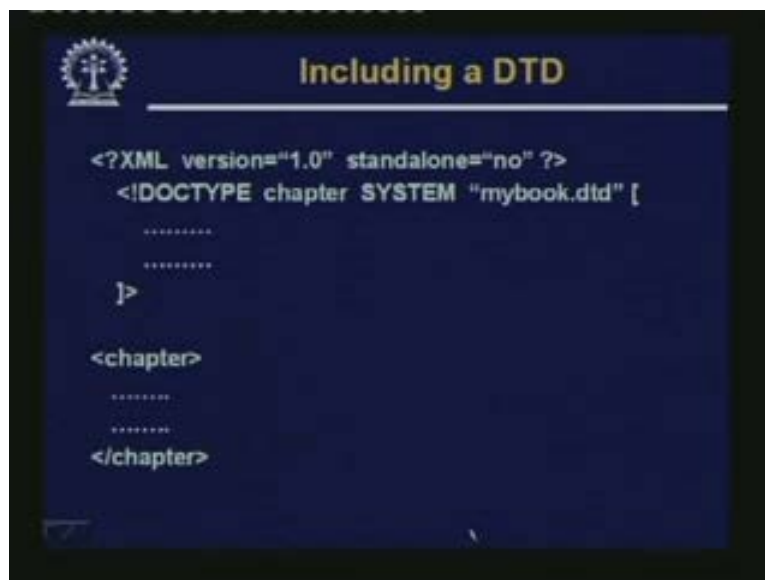


**Issue of White Space**

- By default, white space in a XML document is not significant.
- We can change this:
  - The special attribute `xml:space` can be used to specify that white space is significant.
    - On any element which includes the attribute specification  
`xml:space='preserve'`  
all white spaces would be significant.

Another issue is white spaces. Like html, white spaces are ignored by the html processor. But however there is an attribute XML colon space using which you can make white space significant. See on any element where you want to make it significant. For example a paragraph, you are defining a paragraph on the element that corresponds to the paragraph you include this attribute. XML space is equal to preserve. If you give this then within the paragraph all spaces will be preserved other wise all spaces will get ignored. So this is a way of you can say retaining white spaces.

(Refer Slide Time: 48:40)



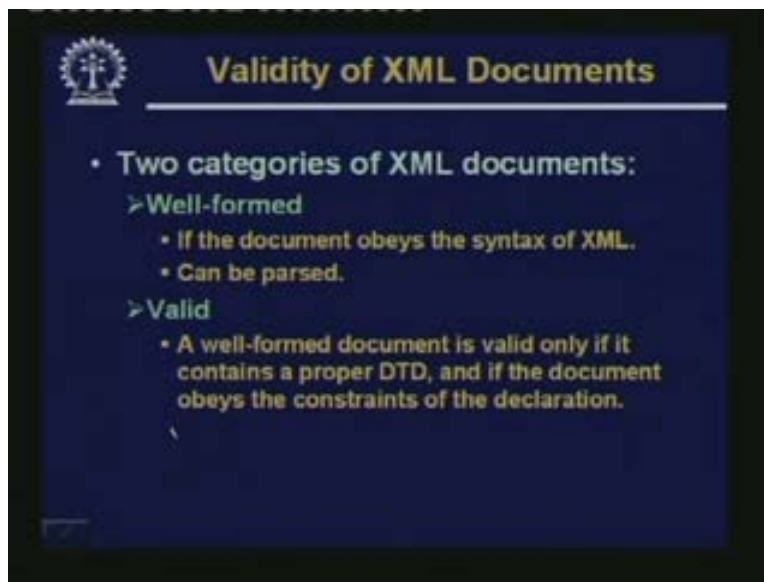
**Including a DTD**

```
<?XML version="1.0" standalone="no" ?>
<!DOCTYPE chapter SYSTEM "mybook.dtd" [
.....
.....
]>

<chapter>
.....
.....
</chapter>
```

Sometimes we may want to include a document type description definition along with an XML document. There is a way to do this first is that in the prolog you specify that this is not a standalone document. Stand-alone equal to yes is the default. You say that this document depends on some other document for proper interpretation. So this is not a standalone document. So standalone you say no and here you specify the corresponding document type definition DTD. This is the way to specify this doctype chapter system this mybook.dtd is the file where all the description of elements and attributes are present. So all the definition you have put there in one place in a separate file and in this XML file you are including them instead of repeating the definition every time in every XML you can simply include them as a DTD. So this is the syntax and after this your chapter type gets defined just like a document type. This is a document type of chapter. So now we start with begin chapter, end chapter, in between whatever tags you have defined you can use them in between. This is just an indication of how you can use a DTD.

(Refer Slide Time: 50:15)



Now XML documents can be categorized as well formed. If it obeys all the syntax of html of XML and if it obeys the syntax it can be parsed. You can have another level of you can say well form ness you can say valid. Valid says it is it will be formed no doubt. But it will also contain a proper DTD and the document follows all the constraints of the DTD. So it is something more just the syntax. Some semantic is also included then you call it a valid document.

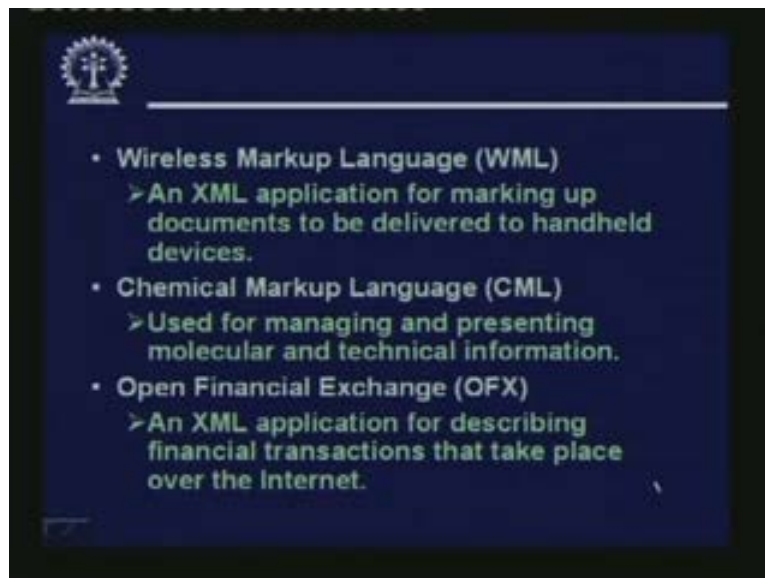


(Refer Slide Time: 51:03)



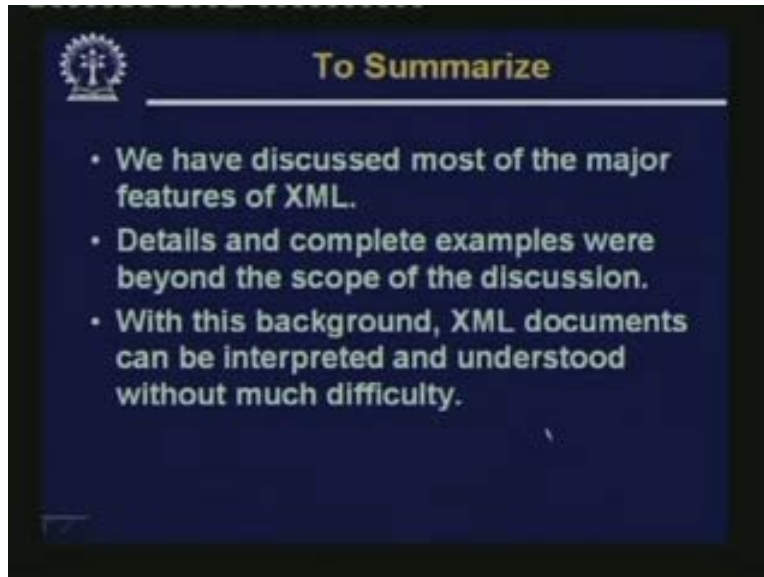
Now standard XML languages there have been many attempts. As I told you that XML is used not only for creating images or documents for web pages but for many other application. Some I am just telling very quickly Synchronized Multimedia Integrated Language SMIL. These have been used for combining audio video text in a synchronized fashion for some applications. Scalable vector graphics SVG this has been used for two dimensional graphics specification. Mathematical markup language MathML. This has been used for capturing mathematical equations.

(Refer Slide Time: 51:42)



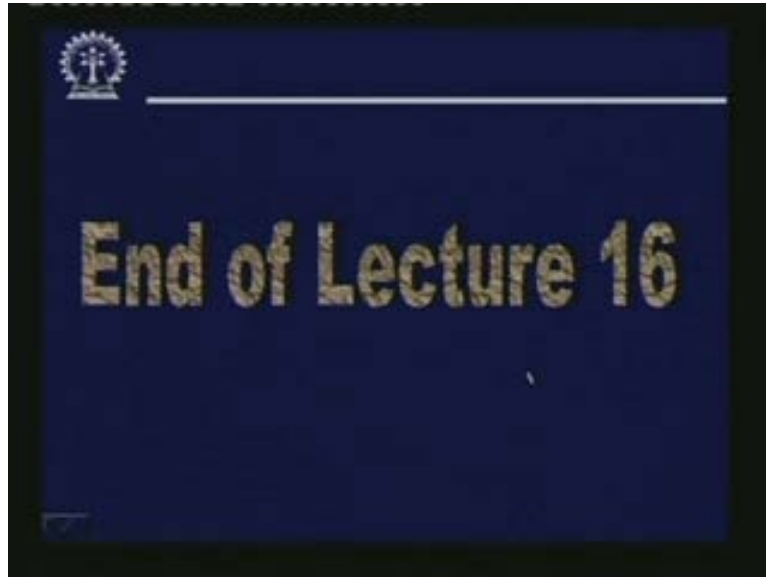
Wireless markup language is very popular. This is used for handheld devices like PDAs and cell phones. Chemical Markup Language CML used for specifying molecular structures. And open financial exchange OFX is used for describing financial transactions.

(Refer Slide Time: 52:05)



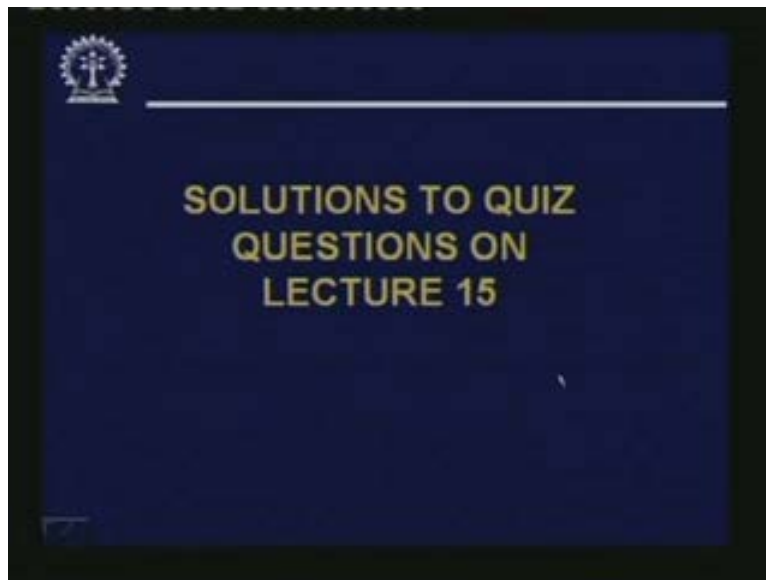
So you can see that there are so many different applications and in this lecture we have tried to discuss just touch on the basic features given some examples to tell you. That means, what are features and how they can be used in an XML application. But in reality XML is much more than that. XML is an open language depending on your application. You can use it in whatever you can.

(Refer Slide Time: 52:36)



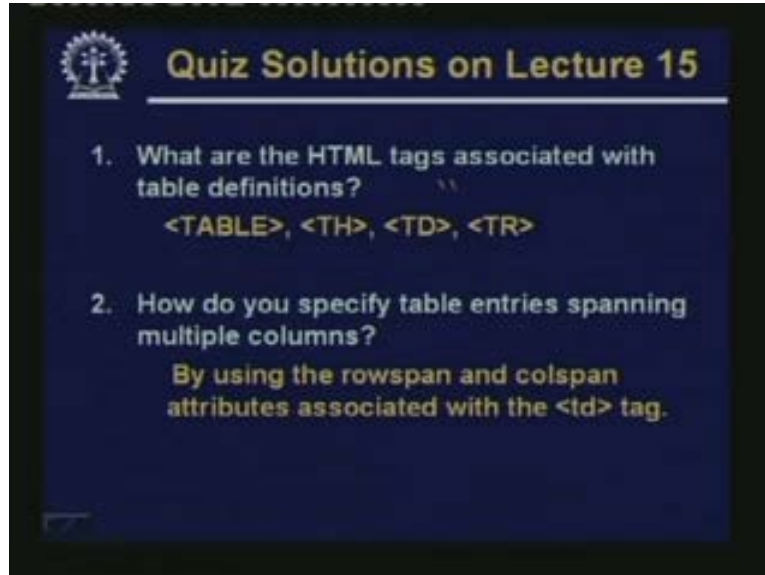
So with we come to the end of today's lecture.

(Refer Slide Time: 52:43)



Let us quickly go through the solutions of the previous lecture.

(Refer Slide Time: 52:45)



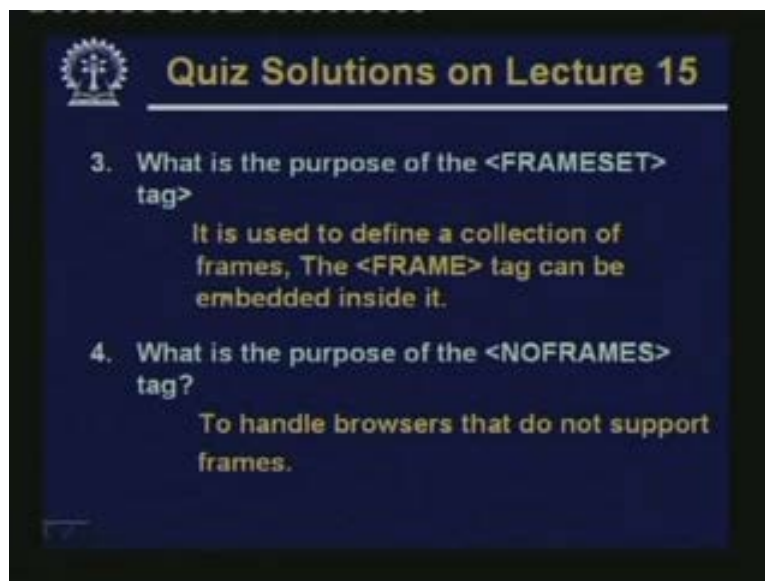
What are the html tags associated with table definition?

These were the four tags table, th, td, tr.

How do you specify table entries spanning multiple columns?

This we have seen using the row span and col span attributes this we can use with td and also th.

(Refer Slide Time: 53:07)

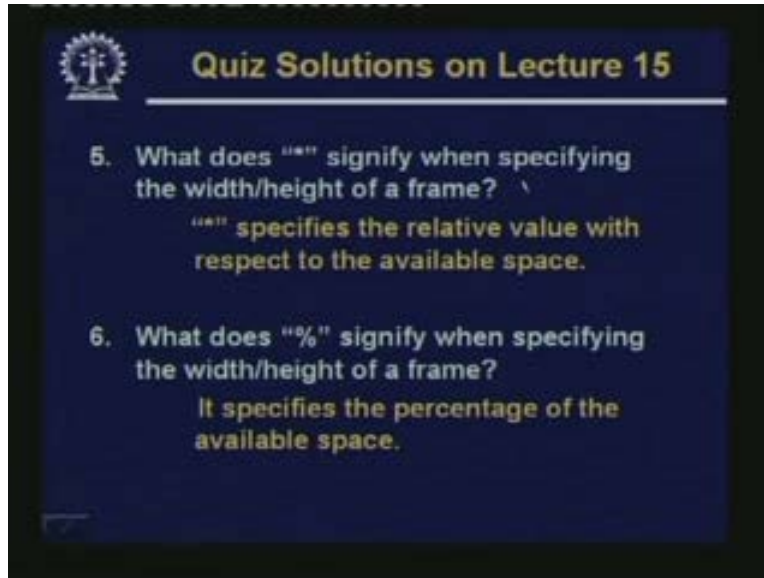


What is the purpose of frameset?

To define a collection of frames and the frame tag can be embedded inside the frame?

Noframes what is the purpose?  
To handle browsers that do not support frames.

(Refer Slide Time: 53:23)



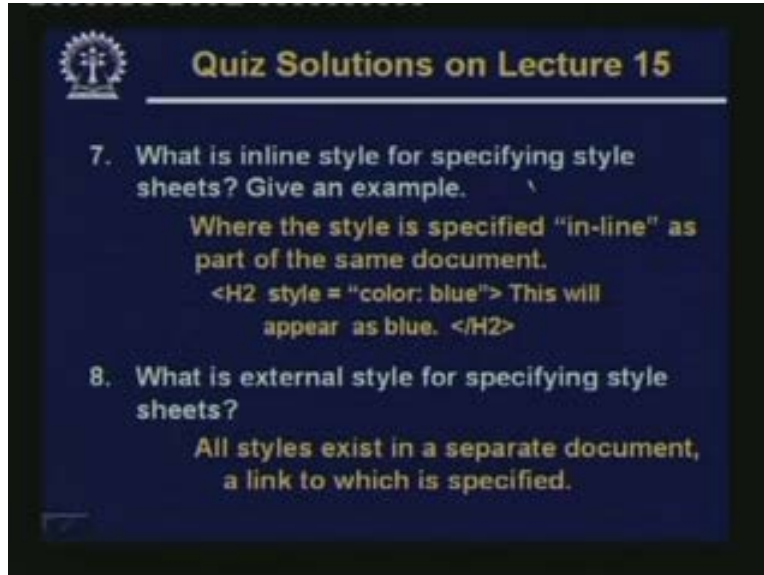
What does star signify?

Stars signify or specify the relative value with respect to the available space. I told you in the previous class there is difference between star, two star, three star. Relatively you can define the width or the sizes of the frames.

What does percent signify?

Percent is the percentage of the available space. The sum total must add up to 100.

(Refer Slide Time: 53:49)



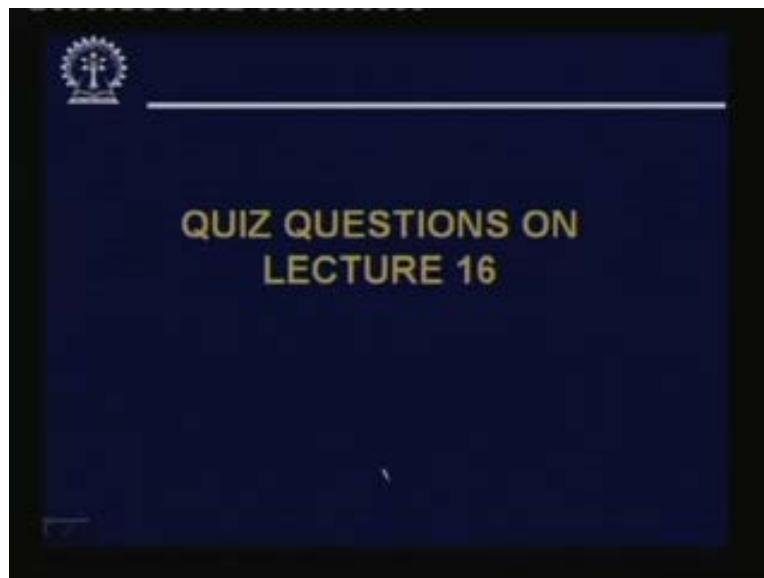
What is inline style with an example?

Inline style is one where the style is specified in line as part of the document. This is one h2. This one example I had given the example.

What is external style?

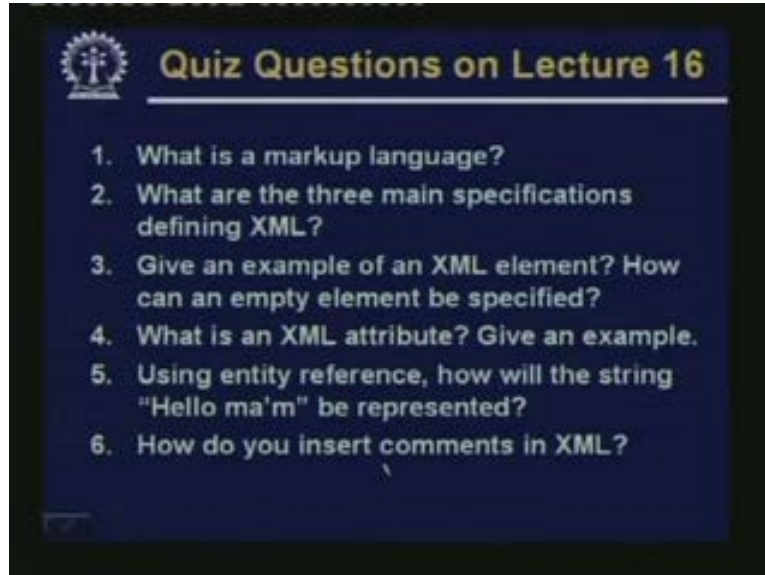
External style is one where the style description are present in a separate document and a link to the document is specified here.

(Refer Slide Time: 54:17)



And some questions for today's lecture.

(Refer Slide Time: 54:19)



What is a markup language?

What are the three main specifications defining xml?

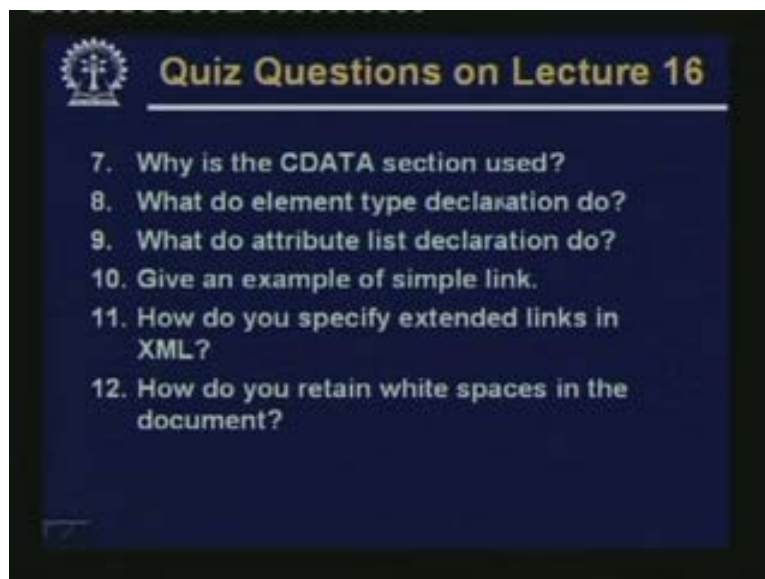
Give an example of XML element? How can an empty element be specified?

What is an XML attribute? Give an example.

Using entity reference how will the string this particular string be represented. \

How do you insert comments in xml?

(Refer Slide Time: 54:48)



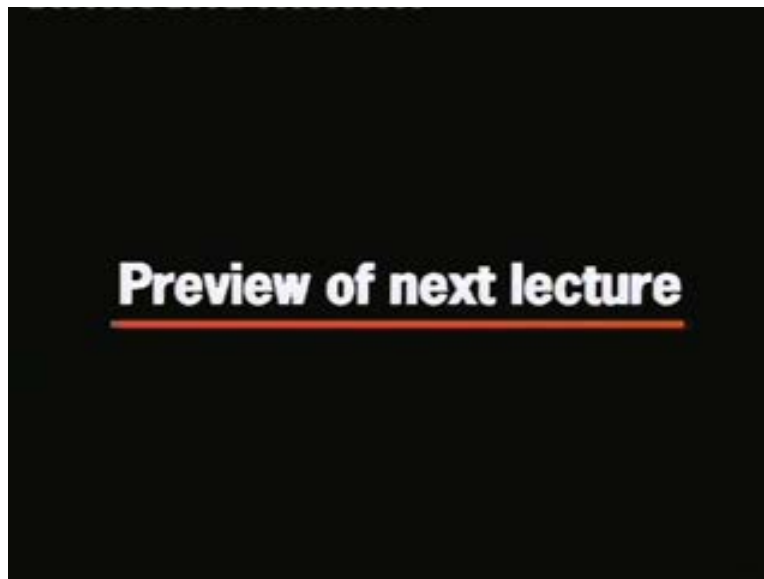
Why is the CDATA section used?

What do element type declaration do?

What do attribute list declaration do?  
Give an example of simple link.  
How do you specify extended links in XML?  
How do you retain white spaces in the document?

Now with this we come to the end of the present module on the creation of web pages. In the next few lectures in our next module we would be talking about how we can create some thing called interactive or dynamic web pages or web content. Thank you.

(Refer Slide Time: 55:32)



(Refer Slide Time: 55:33)





(Refer Slide Time: 55:36)



In this module that we are going to start now, we will be having four lectures and here we will be talking about the different ways of designing interactive web pages. Here in the first lecture today I will be talking about html forms. The next lecture we will be talking about something called image maps. After that cgi scripts or Common Gateway Interface followed by some other technologies which are also similar and competing. First let us try to understand what is meant by an interactive web page see normally the conventional webpage whatever we see, we type in a URL or click a text which points to a URL. The page from the URL gets fetched and it is displayed on my screen.

I am sure all of you are familiar with the search engines people use. Means, we need something I do not know where it is I use the search engines Google, yahoo. These are very popular search engines search engines today. Now what I am trying to say is that, suppose I give something for search to Google, suppose I say I want to search for html. So there is a small white box I type html, I press search and after some times Google gives me a big list of places where I can go for more information about html. Now what I see on my screen now is a page which Google has sent to me. That page contains a number of information all containing information about html links to html.

But is it the case that this page was stored somewhere in the Google server and after my request that page had been sent to me. See this is not possible. Because what the user will be giving for searching that Google cannot predict. There can be millions of possibilities. I can give searching html, I can give hypertext markup language, I can give html form I can give so many things. So a more logical approach would be that Google stores this somewhere in a database. Depending on the users requests that get fetched from the database the information and they get dynamically formed into a html document and sent it back to me. This is what interactive page means. That the page is not residing as it is on the server. But depending on the request it gets created and it is sent back.