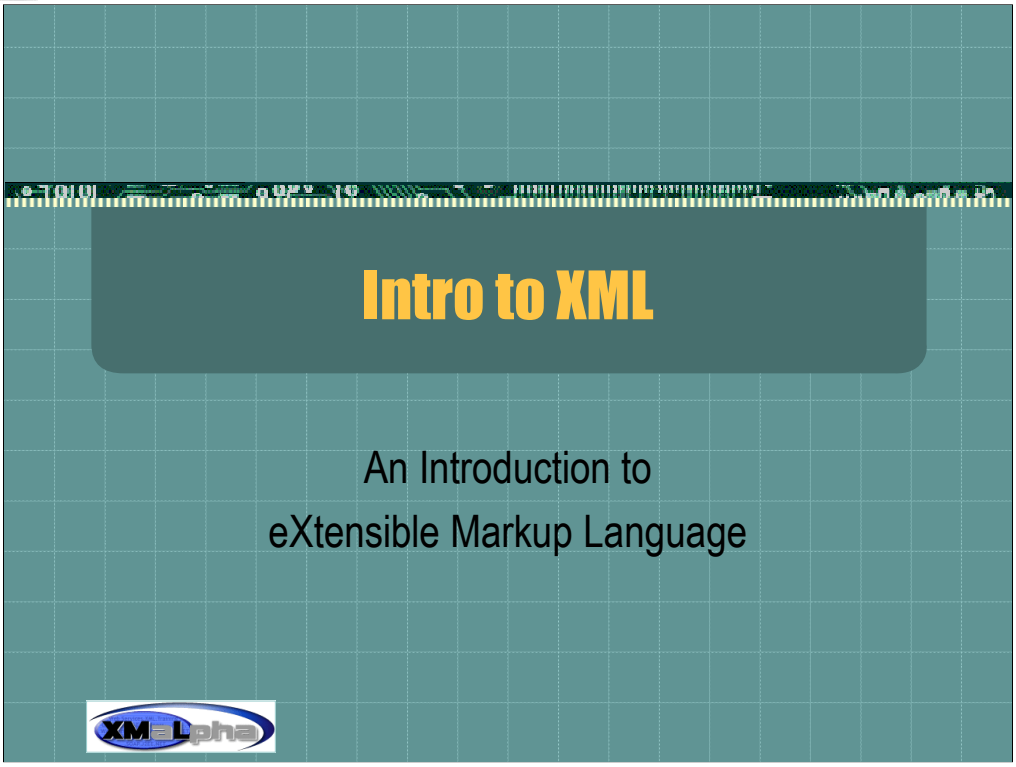


Welcome to: **Introduction to XML**

XMaLpha Technologies, LLC.,

http://XMaLpha.com

INTRODUCTION TO XML (HANDS-ON)

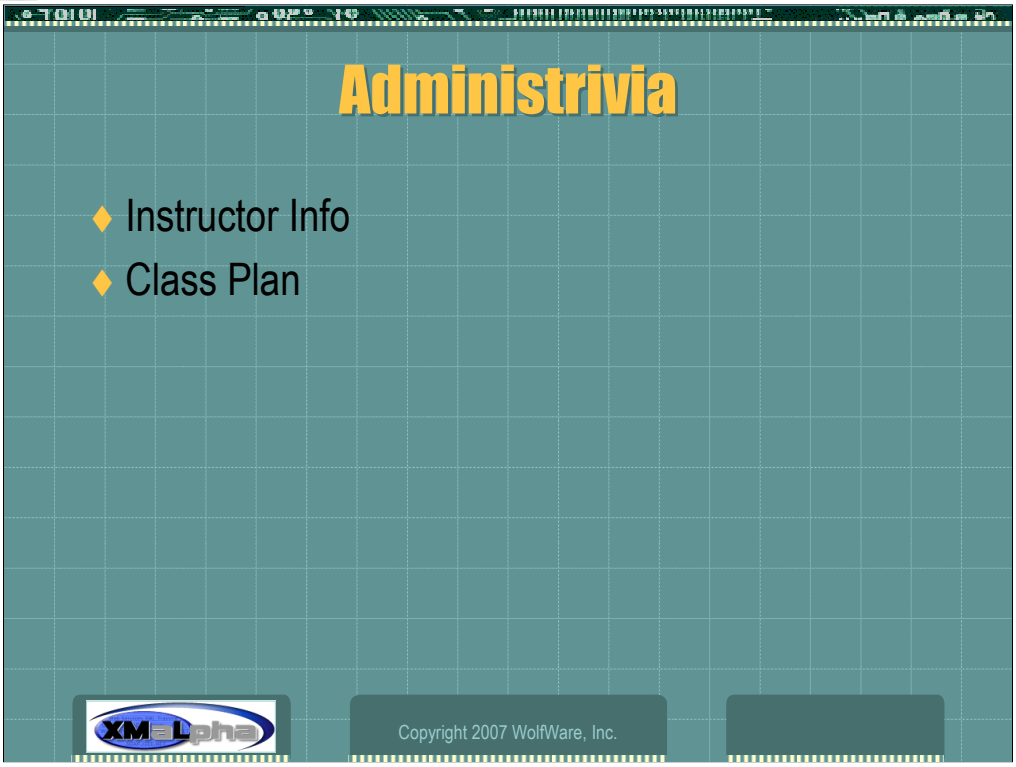
Technical

This full-day session provides a comprehensive introduction to the major topic areas in XML technology, and provides an introduction to Advanced XML. Learn what all the fuss is about and how to jump immediately onto the fast-track for XML development.

Instructor: Joyce Deeb,

Senior Consultant, XMaLpha Technologies, LLC

Joyce Deeb is a senior consultant with XMaLpha Technologies (http://XMaLpha.com), has considerable experience in software engineering, high-level application design and development, advanced computing techniques, strategic technology planning, curriculum development, and teaching. Her level of technical depth in application programming, Java & XML development, web-based applications, data warehousing, knowledge-based systems, and parallel processing is impressive.



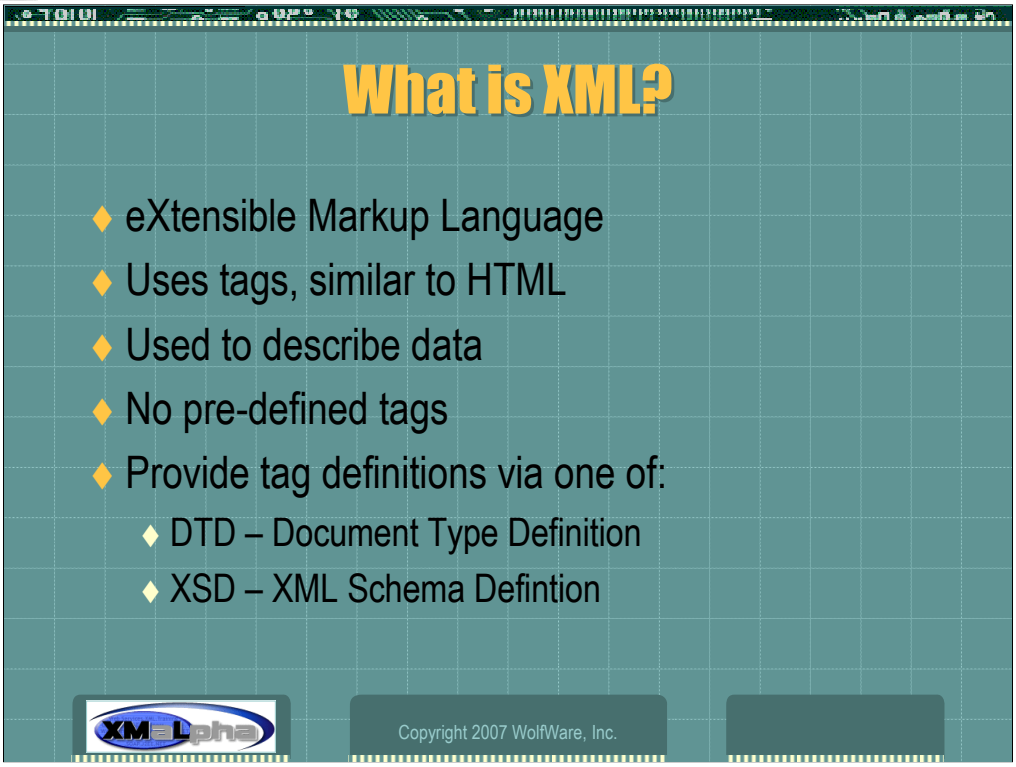
Class will be a mix of lecture, group exercises and hands-on exercises.

XMaLphaTechnologies
Consulting & Education

"Got Meta-Data?"®

XMaLpha Technologies is a premier provider of full-scale, industrial-strength, XML solutions. As experts in analysis, design, and implementation, the talented consultants at XMaLpha understand data integration using XML. Learn how XML can provide a totally extensible, easy-to-learn, and richly featured universal format for structuring data and documents that can be exchanged efficiently over the Web using .NET, Java, and other interoperable technologies. Whether your needs call for business-to-business solutions, sophisticated Web Services, Content Management Systems, end-to-end integration with legacy data, structured dynamic content generation, or education and knowledge transfer on the latest XML, Java or .NET technologies, XMaLpha can help.

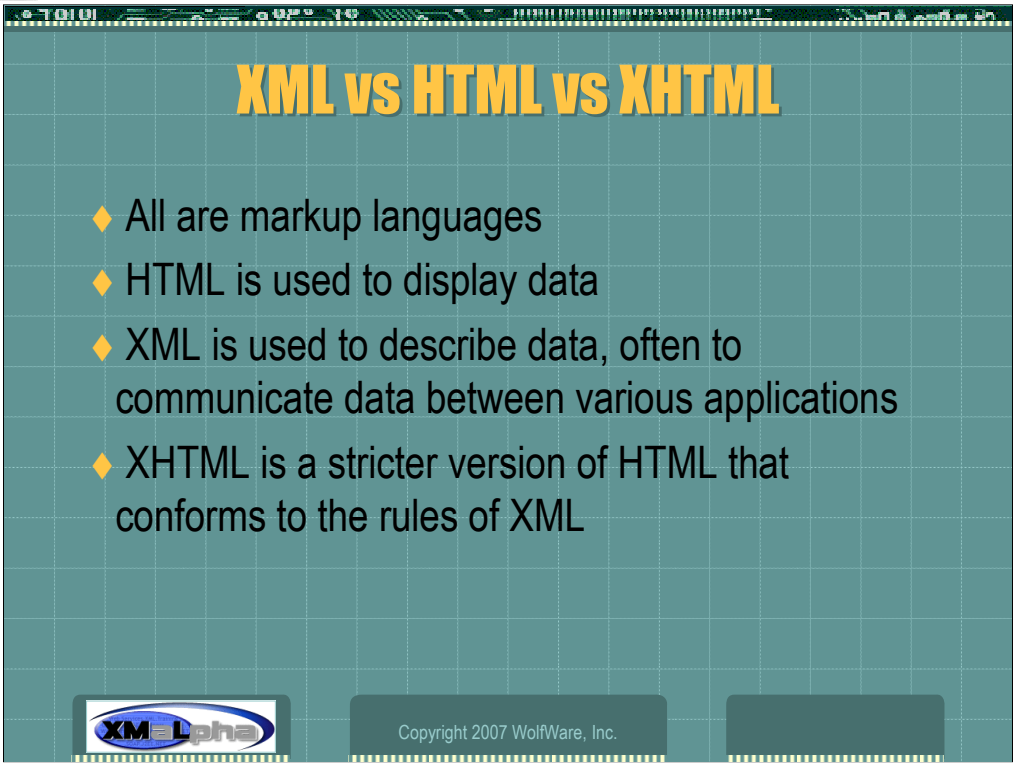
http://XMaLpha.com



Not a language like we think of programming languages.

It is a markup language, and hence uses tags, but there aren't any predefined tags.

It is used to describe, or communicate, data between different endpoints. However, each application of this technology must define tags specific to that application. This is done by defining either a DTD or a schema to accompany the applications XML documents. This DTD or schema describes rules for the XML document. We'll see more on this later.



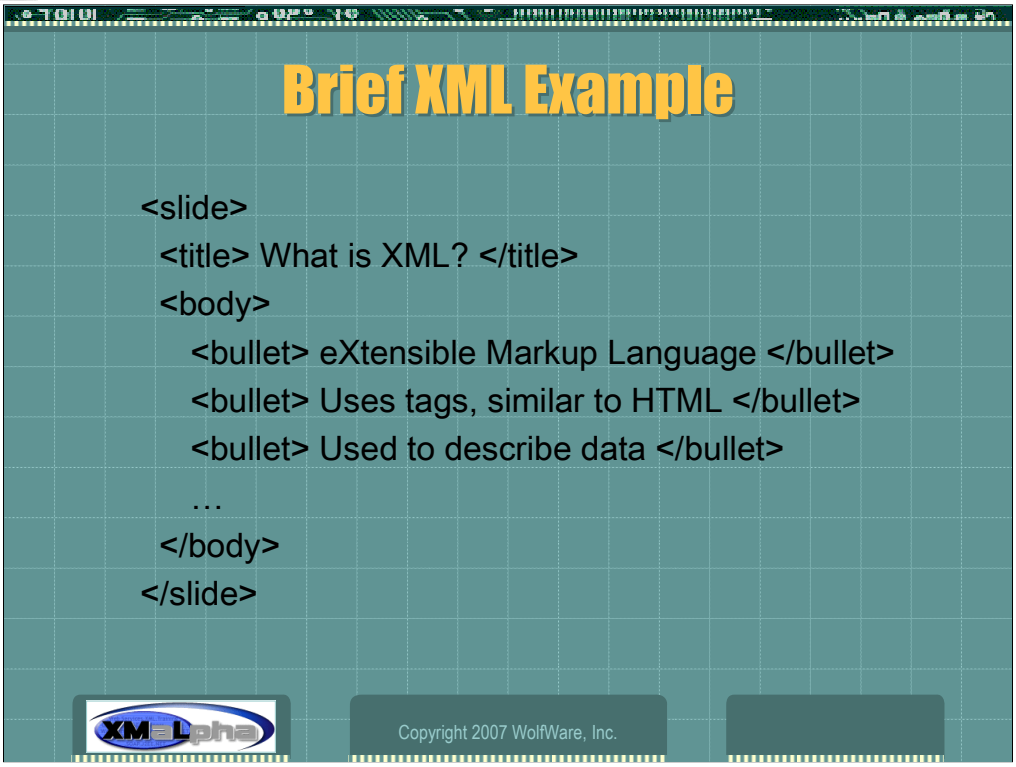
Inevitably, XML always gets compared to other markup languages, especially HTML.

There's not much to compare. They are both markup languages in that they use tags embedded in a document.

However, HTML is used to display data. The tags tell browsers how to format and display the contained data.

XML is used to describe data, typically for communication purposes where you have data being used by multiple applications or even parts of applications. The tags identify what is contained in a given tag. We'll see an example next.

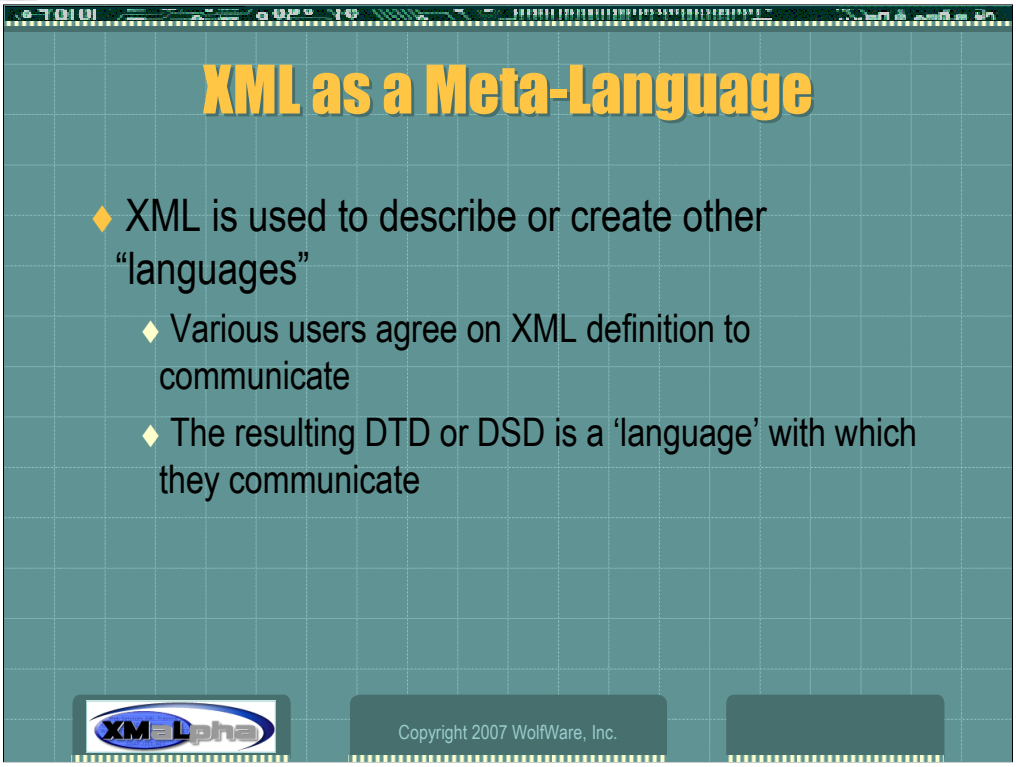
HTML does not always need to be well-formed. XHTML is a stricter version of HTML that conforms to the rules of XML. You could say that it is both HTML and XML. More importantly, it is an example of using XML for a specific application.



This is not a complete XML document, but it gives a general idea of what one might look like.

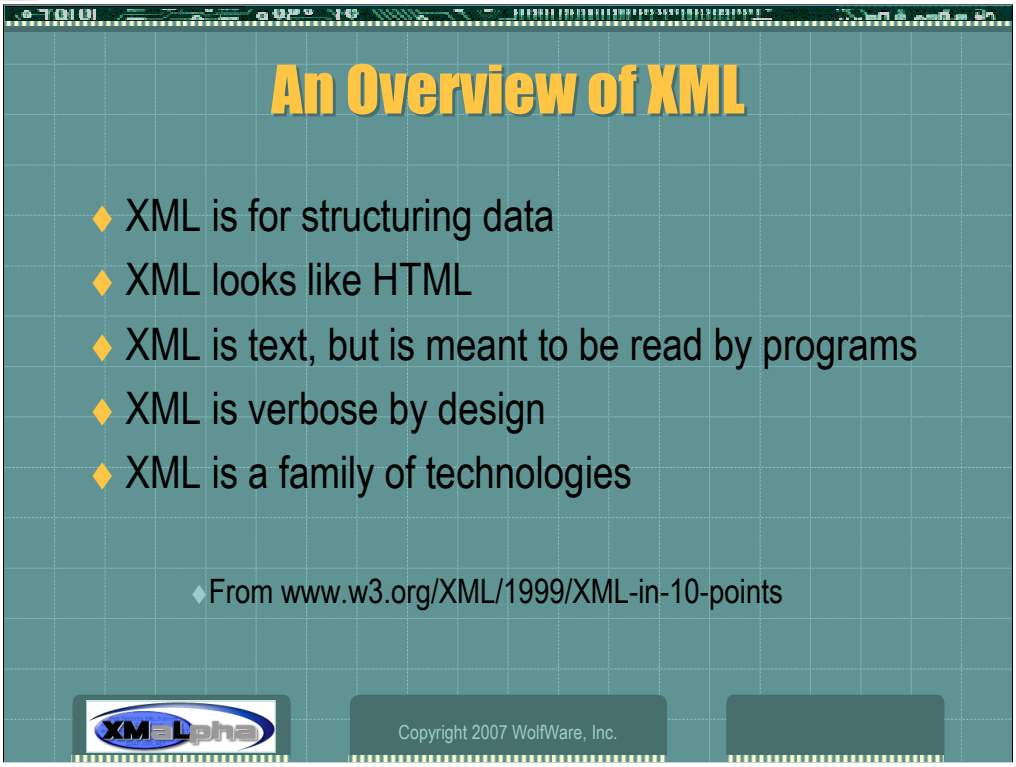
This is one possible XML representation of the previous slide.

Your first reaction might be that this is just another version of HTML, mostly because slides are a display of data. While that is true, and probably what you would do with slides, there are other applications that you might want to do with these. For example, what if you want another application that will take the <title> out of each slide and create an outline. There are other things we can do with these, but they most likely require a more elaborate XML document which we'll learn about later.



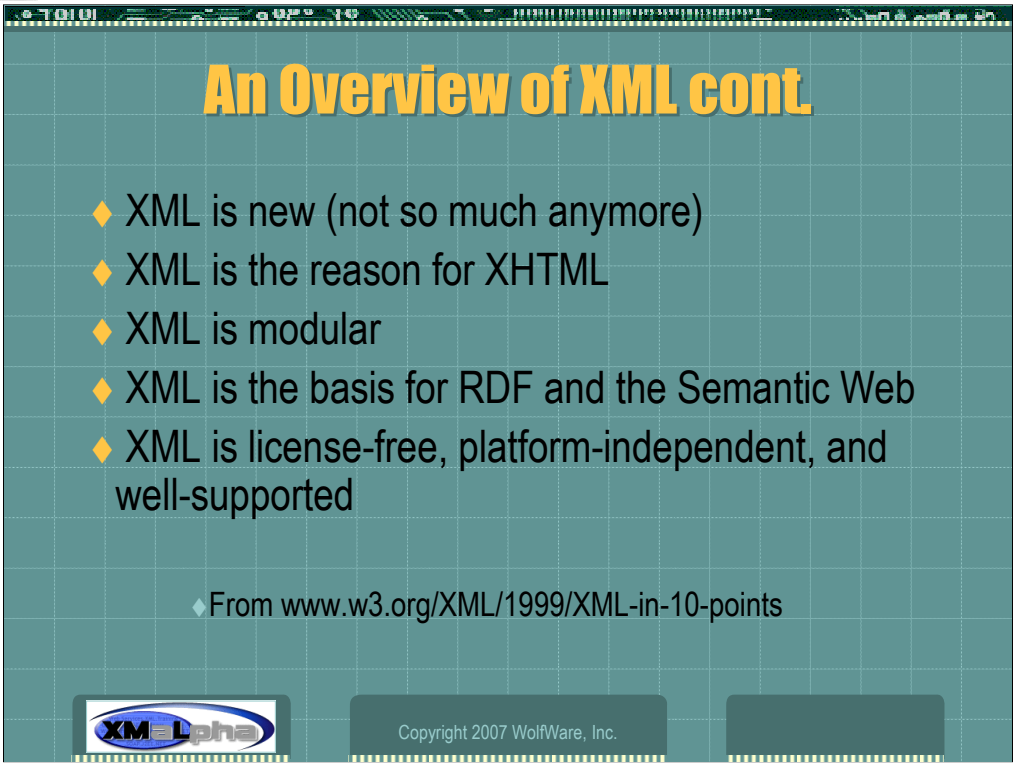
You can think of the M in XML as standing for 'meta', because XML is a meta-language as well as a markup language. XML is used to define a sort of language to be used by end-users to communicate data.

We'll learn about DTDs and DSDs later.



See http://www.w3.org/XML/1999/XML-in-10-points

1. Structuring data – Other examples of structured data include spreadsheets and databases.
2. Resembles HTML – They look alike, but the similarities end there. Except that you could use XML to define HTML for displaying web pages. In fact, that's what XHTML is.
3. Text – Unlike some other structured data which is encoded in proprietary formats, XML is plain text, which can be read by people. However, it is intended to be read by programs that can parse the language and interpret it appropriately.
4. Verbose – a typical XML encoding of data is often larger than the data itself. Think of an XML encoded memo, for example. The tradeoff of storage space is worth the advantages of how versatile the data becomes.
5. Family of Technologies – XML isn't a language, it is a technology or technique. It is not nearly as useful by itself as it becomes when used with other related technologies such as XSL and XSLT. We'll learn more about these later in class.



Continued:

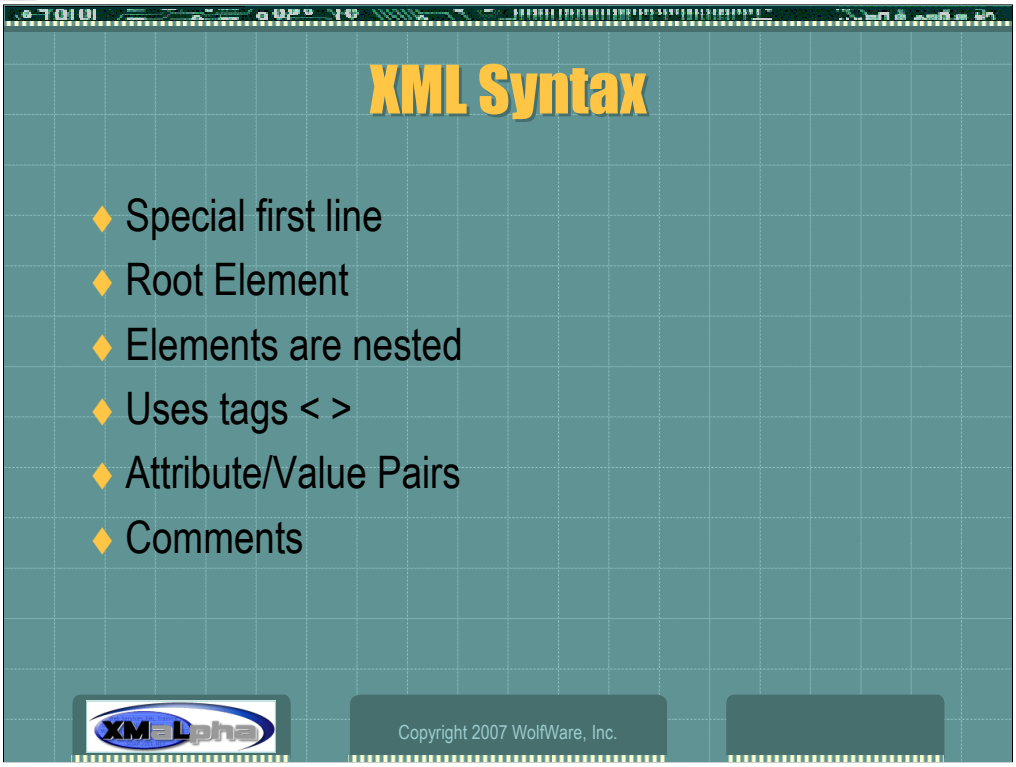
6. New – This all depends on your frame of reference. XML has been around since the mid-90's and is based on technologies that have been around since the early 80's.

7. XHTML – a stricter version of HTML which conforms to XML standards.

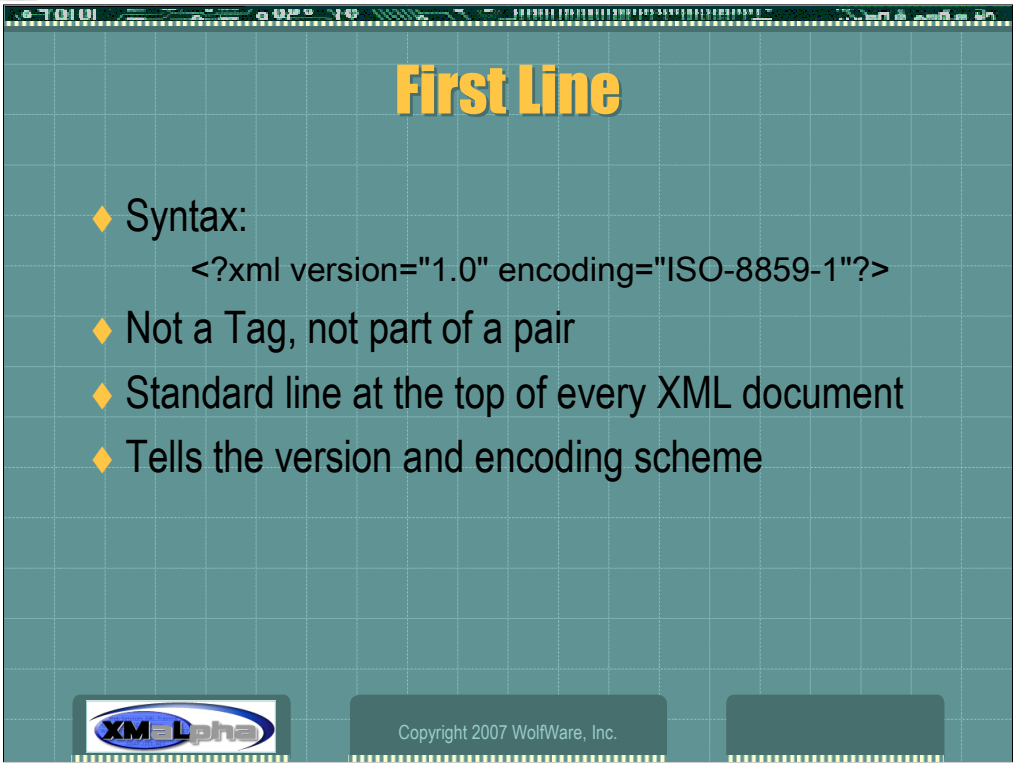
8. Modular – XML definitions can build on each other, and namespaces are used to keep names straight.

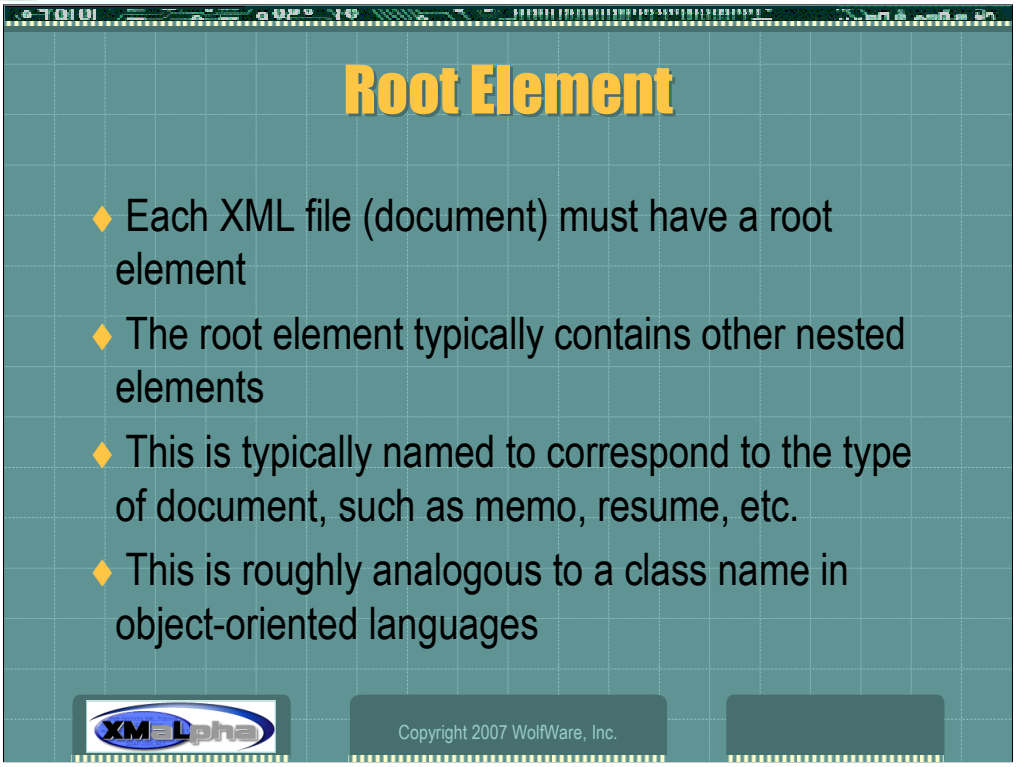
9. Basis for RDF – RDF is beyond the scope of this course, but briefly, it is a means of describing the resources available on the web, hence the term Semantic Web.

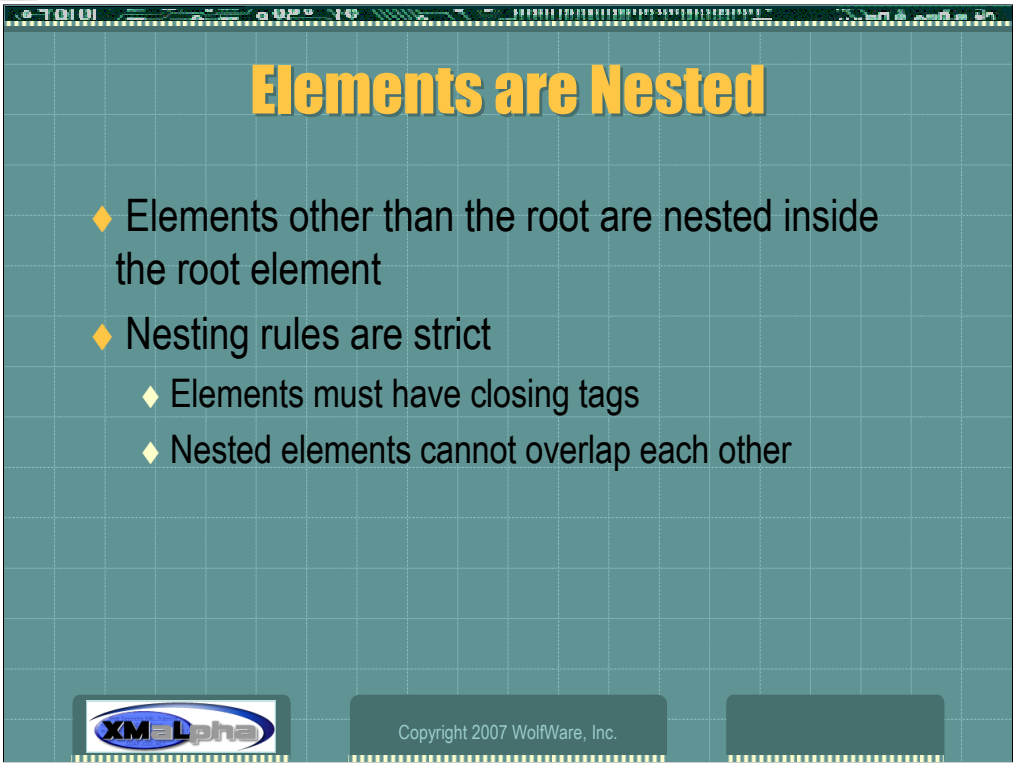
10. Aahhh, the way it should be.



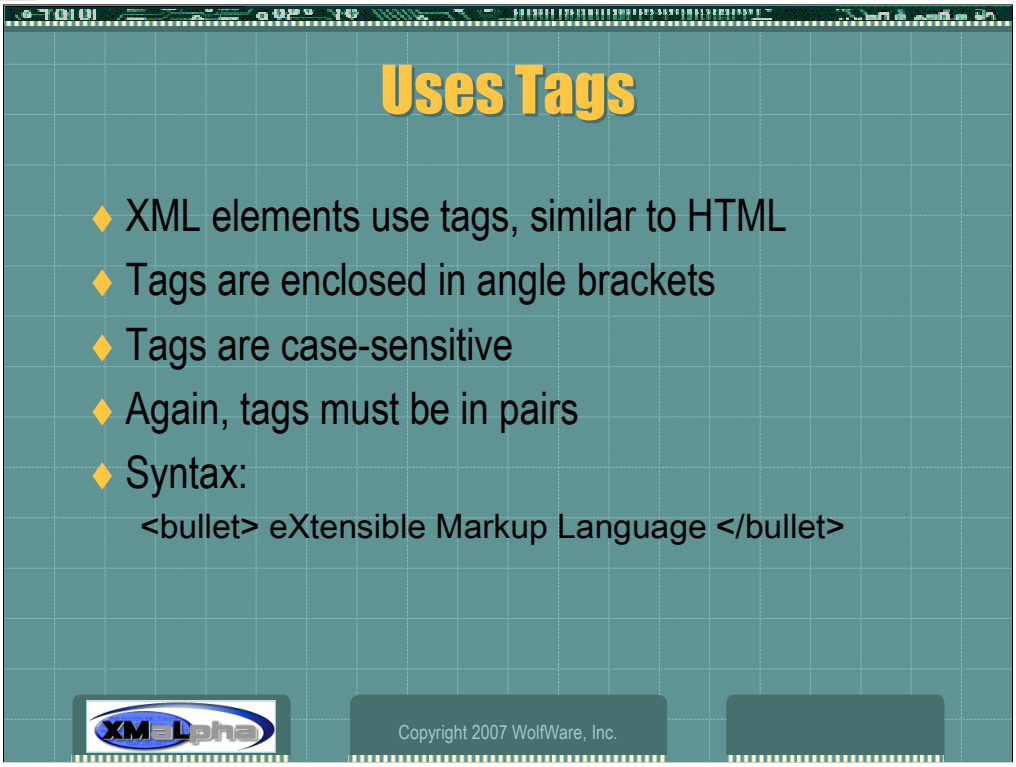
The next few slides will discuss each of these individually.



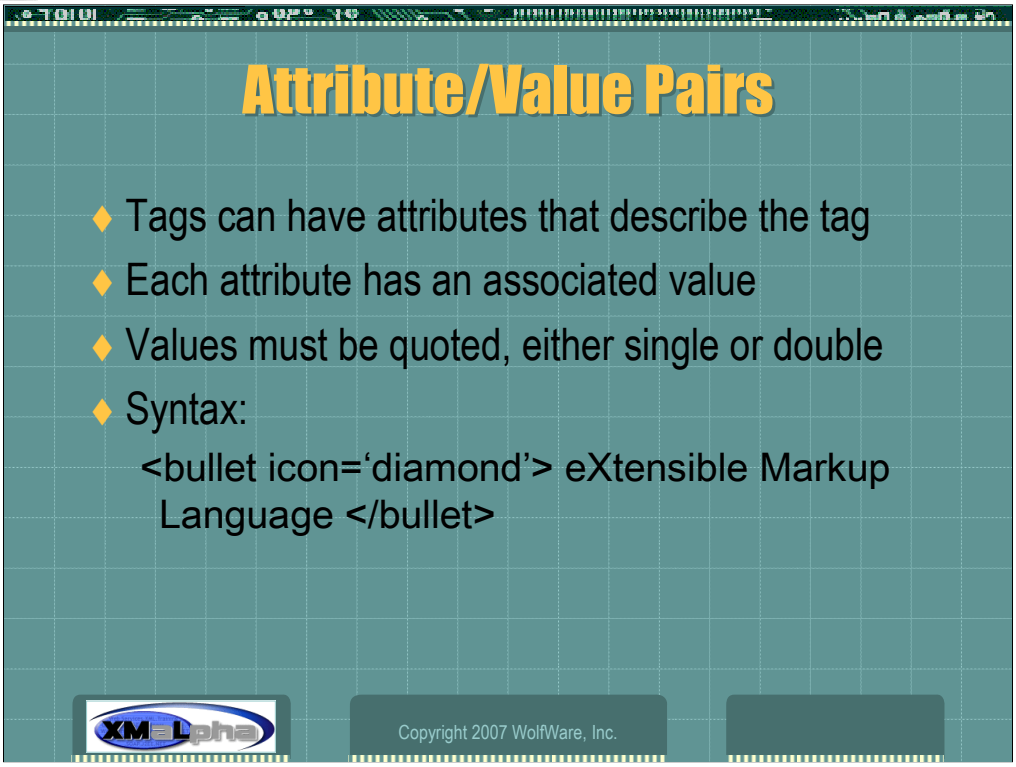




Test – Never let 'lines' cross.



Tags are roughly analogous to data members of a class in OOP.



If you want to continue the OOP analogy, Attribute/Value pairs would be like having contained objects in your class. So the parent class contains objects as compared to primitives.

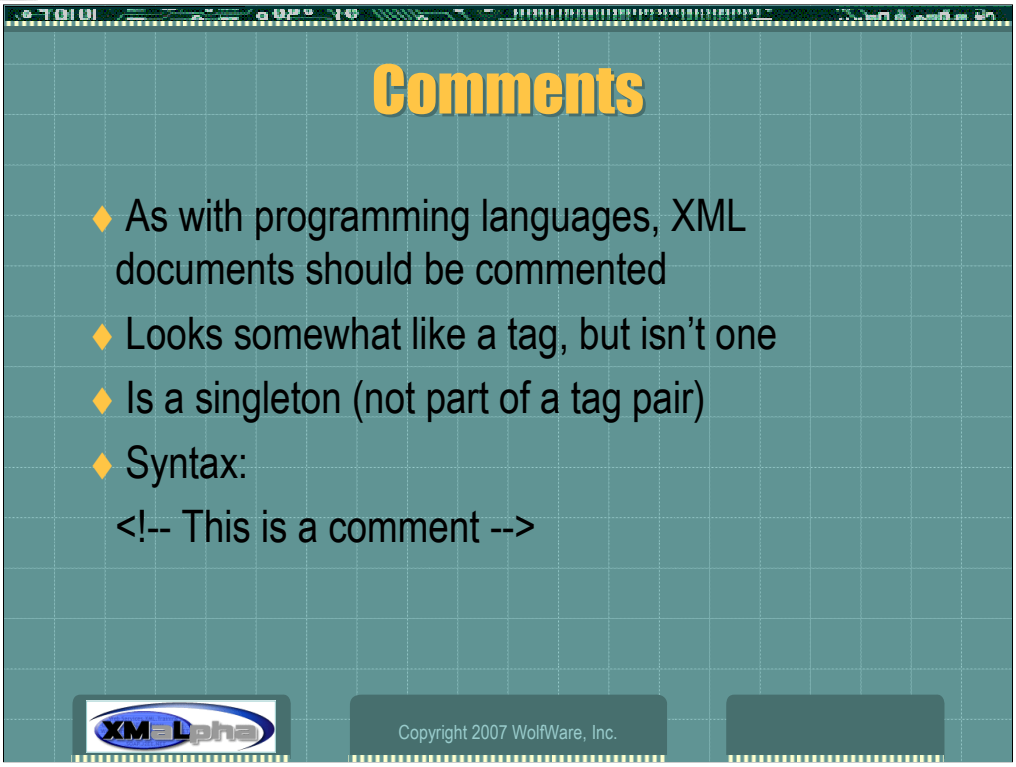
HTML has lots of attribute examples:

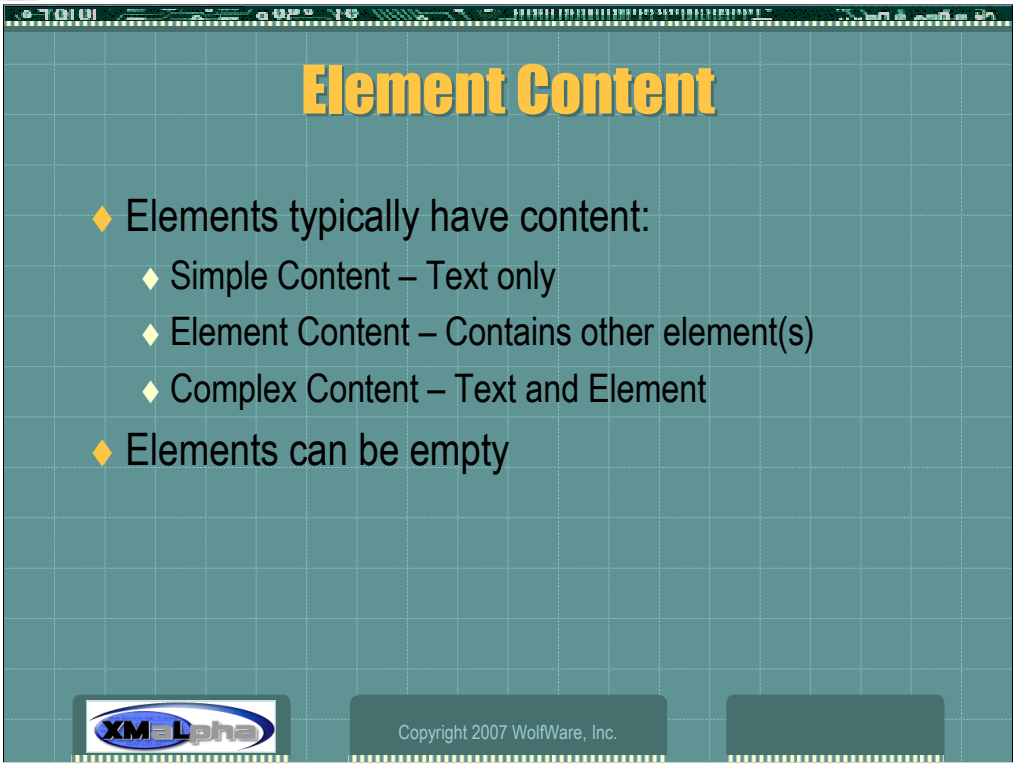
img tag has src attribute

a tag has href attribute

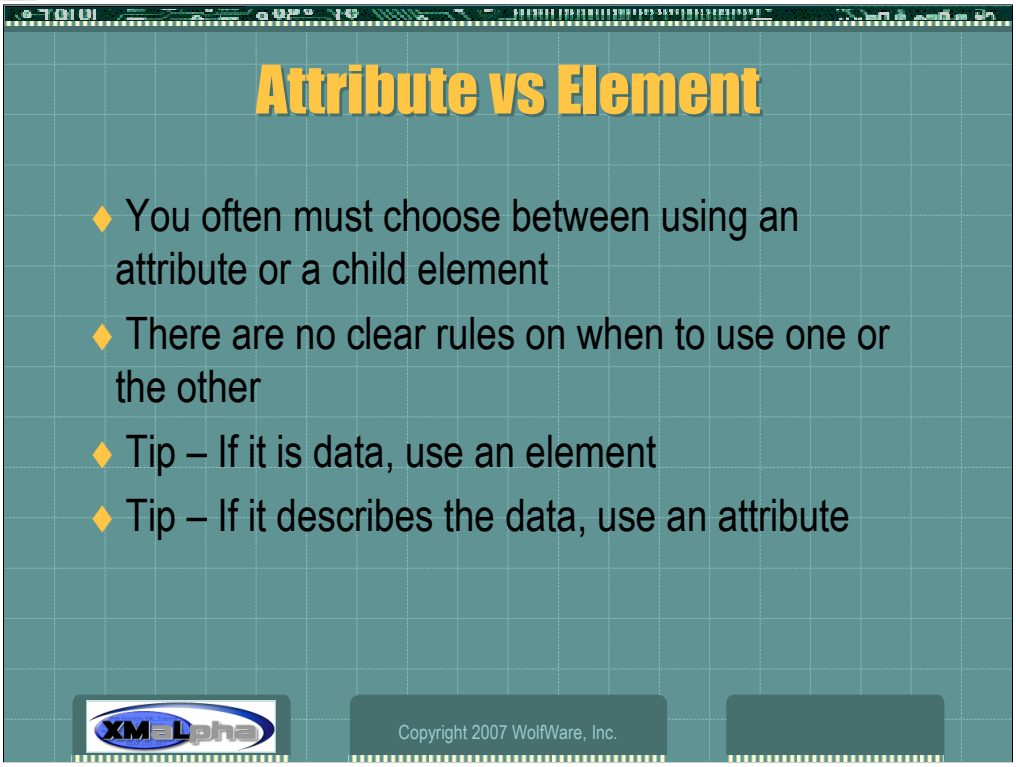
table has cellpadding attribute

etc…





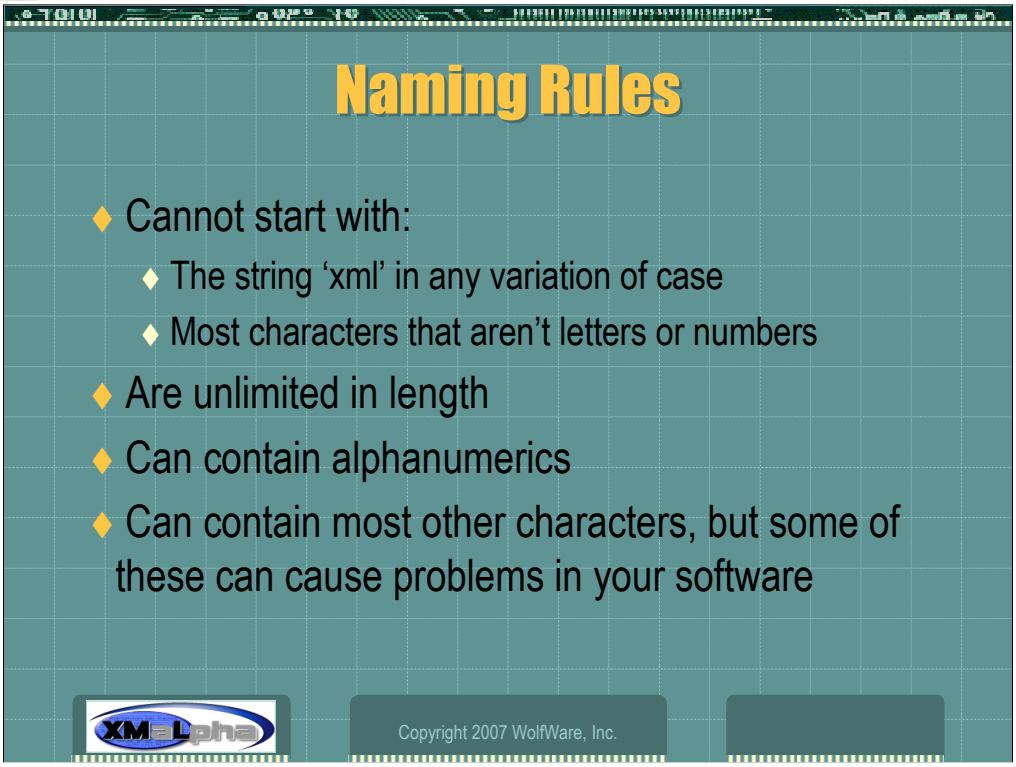
Empty elements can be there as a marker, and often have attributes.

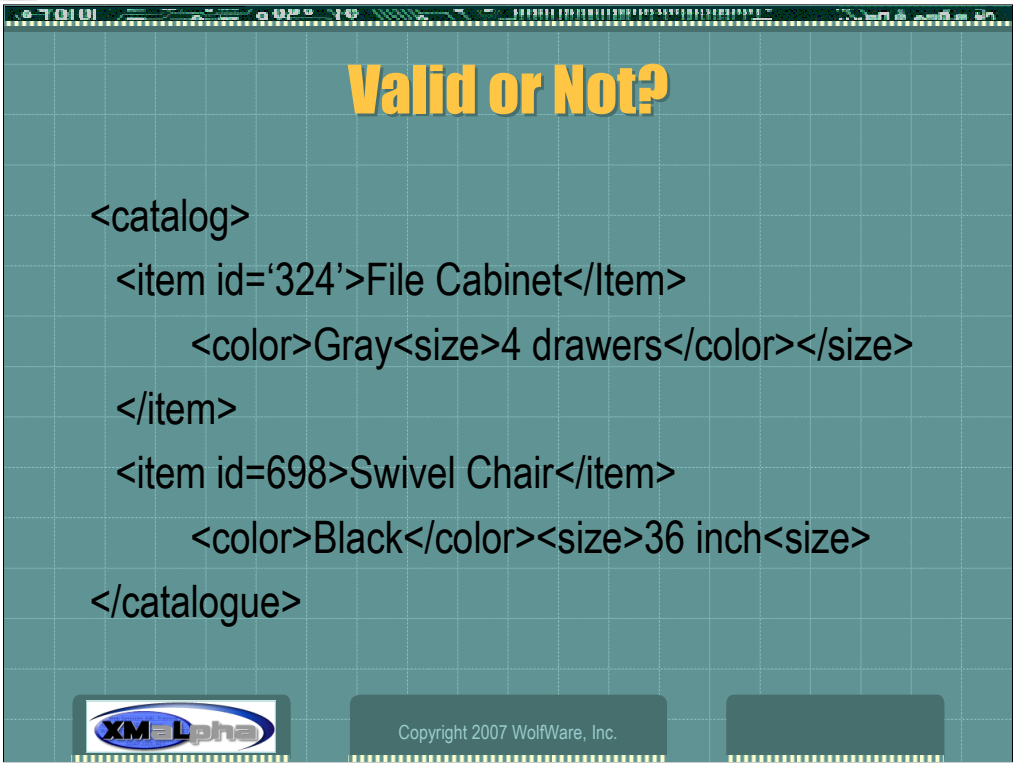


Data – Given a card catalog, things like ISBN, author, title

Attribute – icon in the bullet example

You will see other more elaborate guidelines, but I don't find them very reliable, and can often debate most of them.





Find the errors together in class: (5 minutes max)

Errors:

/Item or item

/color and /size are out of order (lines cross)

Second id is not quoted

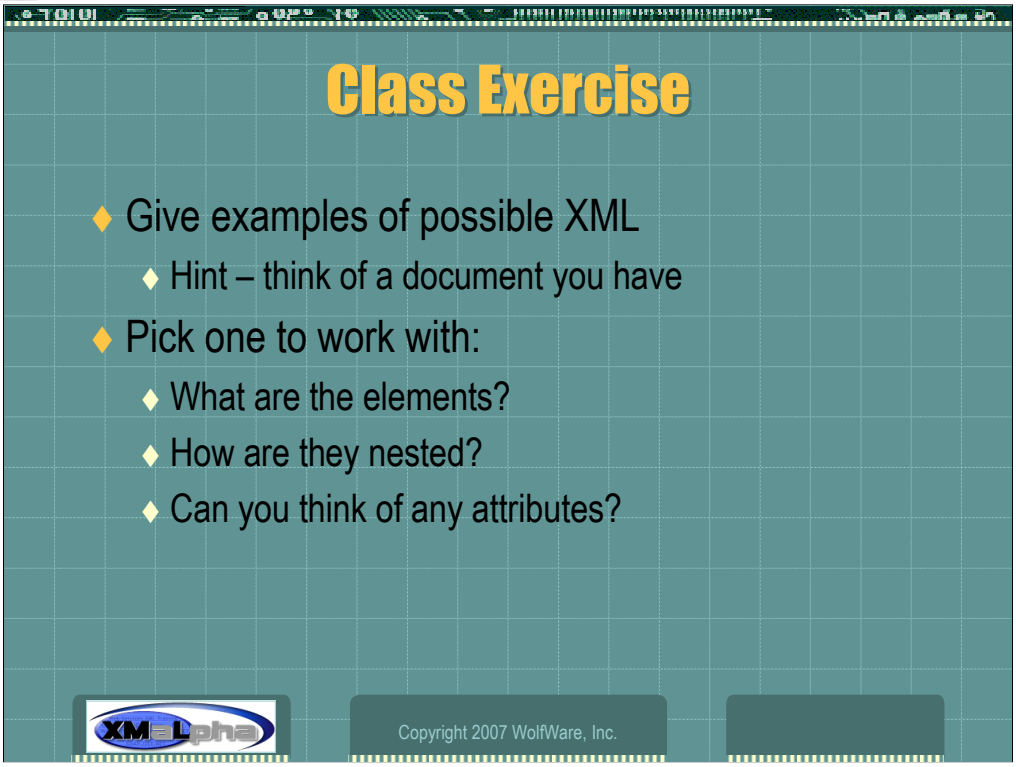
Second size is missing the /

catalogue

This is not a particularly realistic example, but serves the purpose of this exercise.

Although the indentation is not illegal, good form would put color and size on separate lines.

For a catalog, id would typically be a child element since it would likely be part of the printed catalog (data). However, audience might be an attribute if they are printing separate catalogs for the public, suppliers, internal use, etc.

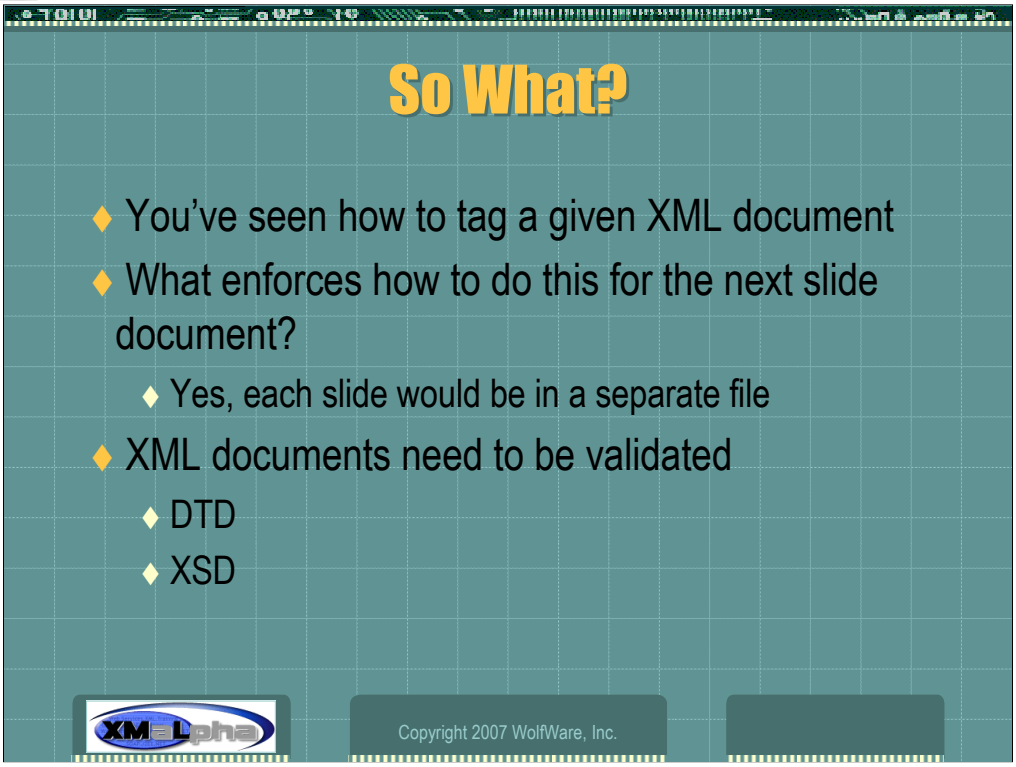


For hands on class – Go through options together. Pick one, then give students 10 minutes to do bullet 2. Go through example together in class to let them see how they did. Total time: 20-30 min.

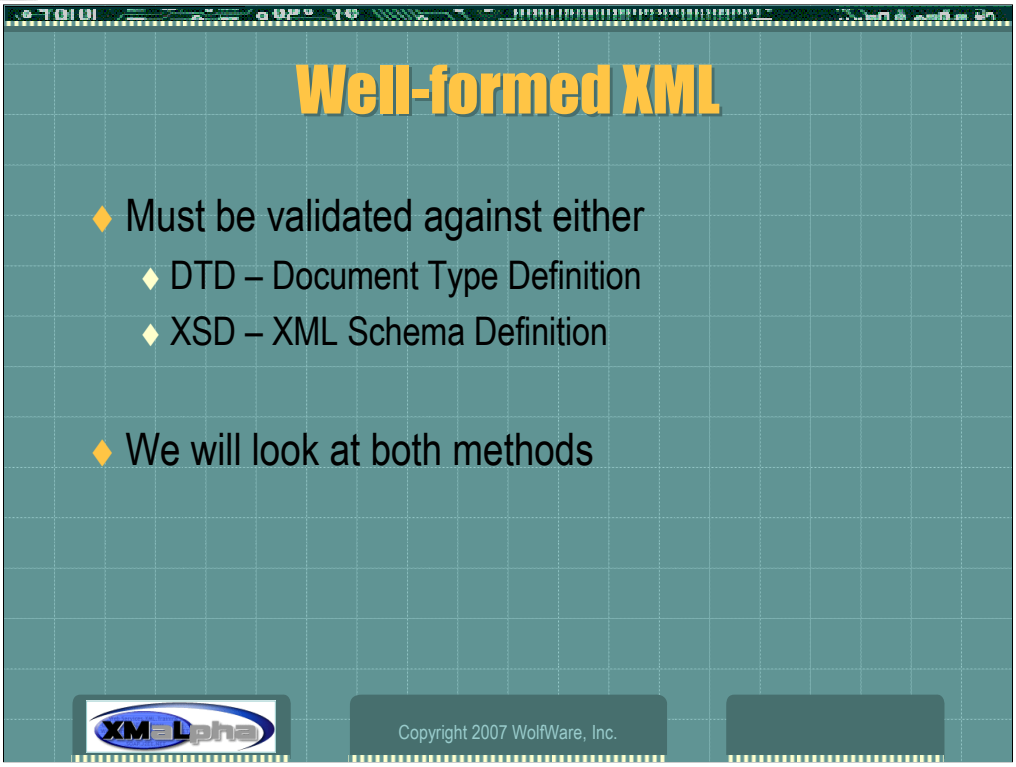
For lecture, can do the same, or go through the entire exercise together. In that case, total time is 10-15 min.

Also, homework:

Provide a document, and have students turn it into an XML document.



All of those syntax rules have to be enforced to get well-formed and validated XML.



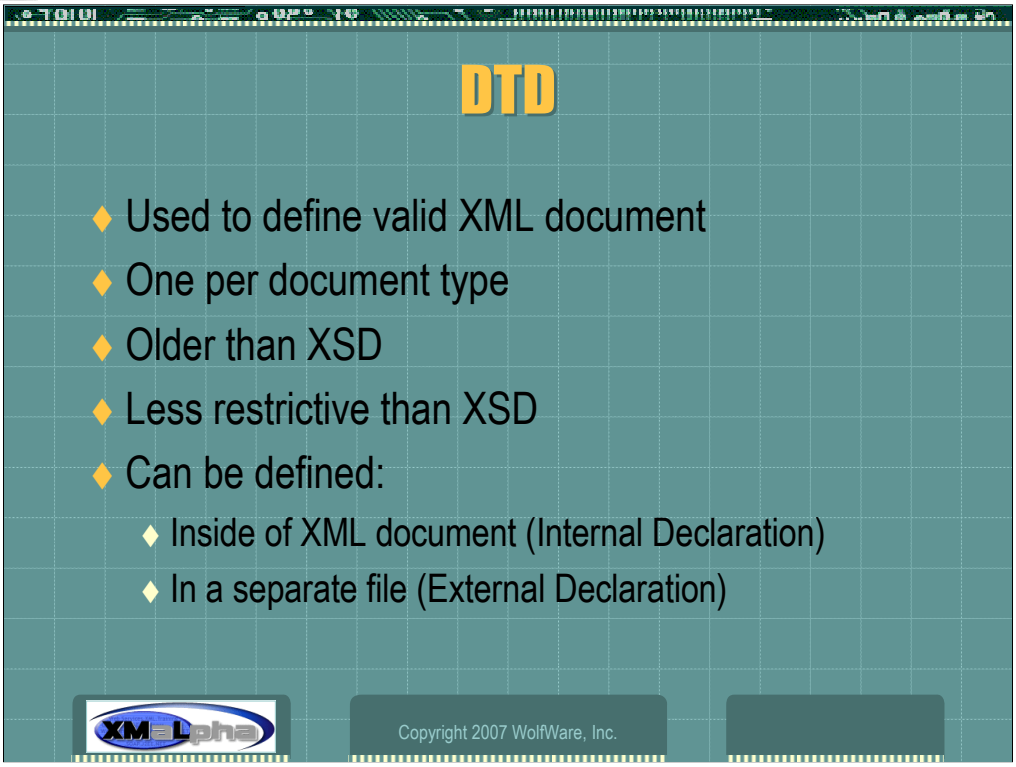
To have well-formed XML document, it must be validated against either a DTD or a schema that describes the rules for that particular type of document.

This is analogous to having an instance of an object correspond to a class definition.

DTD is older

Schema is becoming more popular

Schema allows more enforcement than DTD

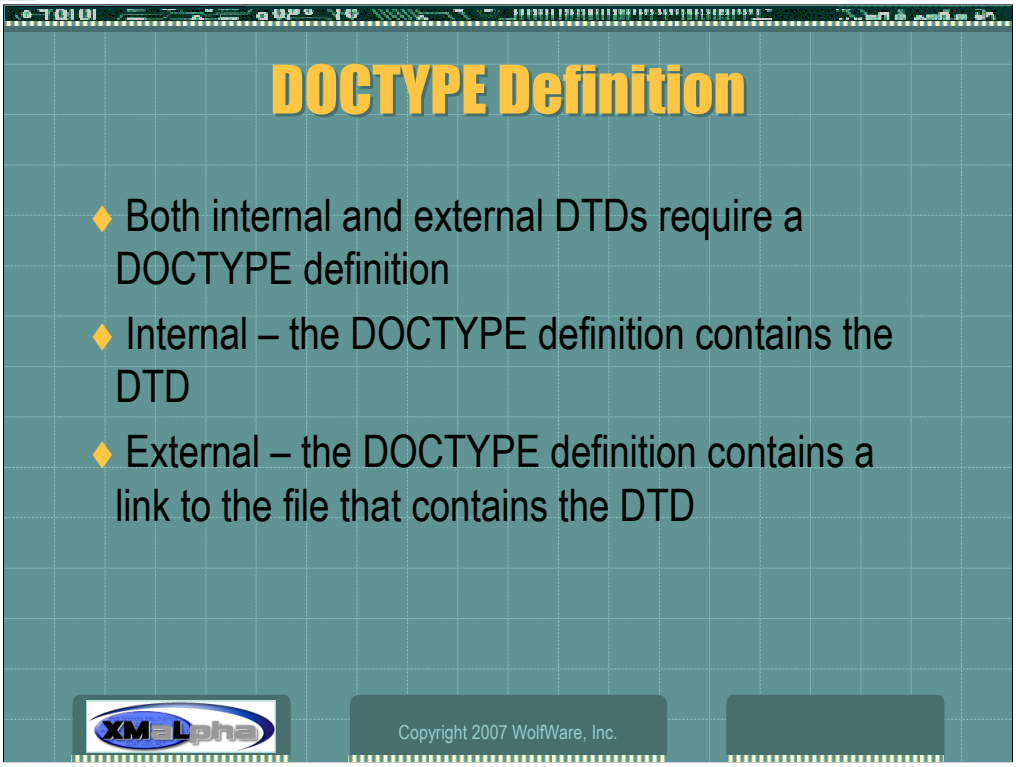


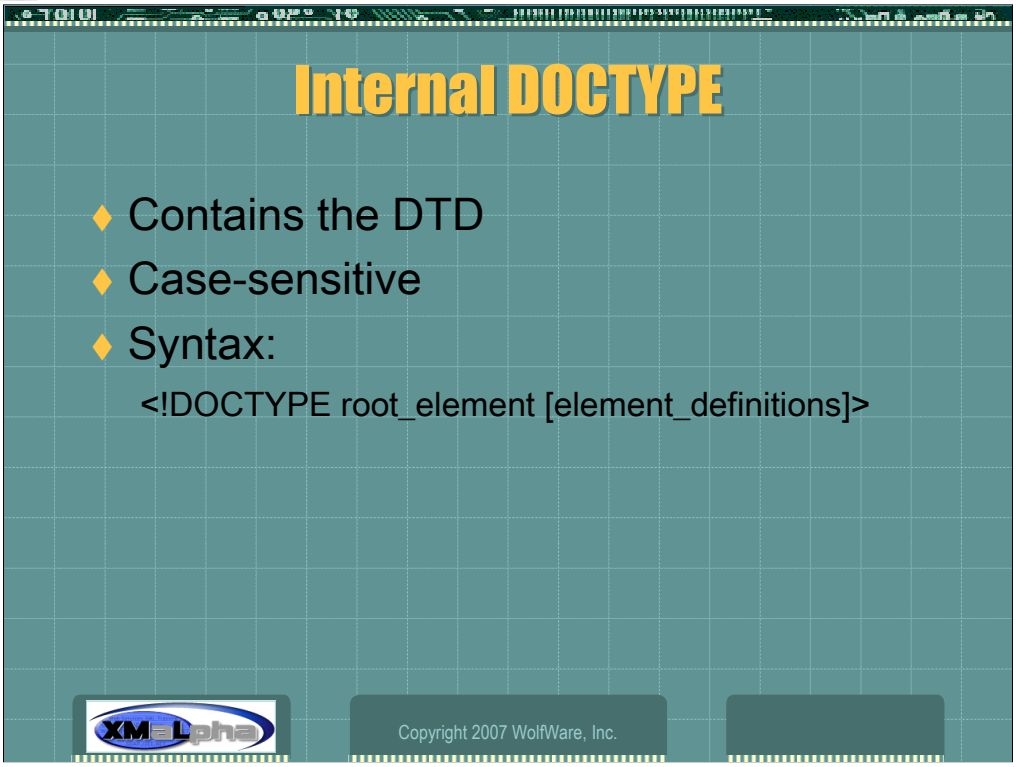
Older, and hence more popular, but becoming less-so, as schemata become more popular.

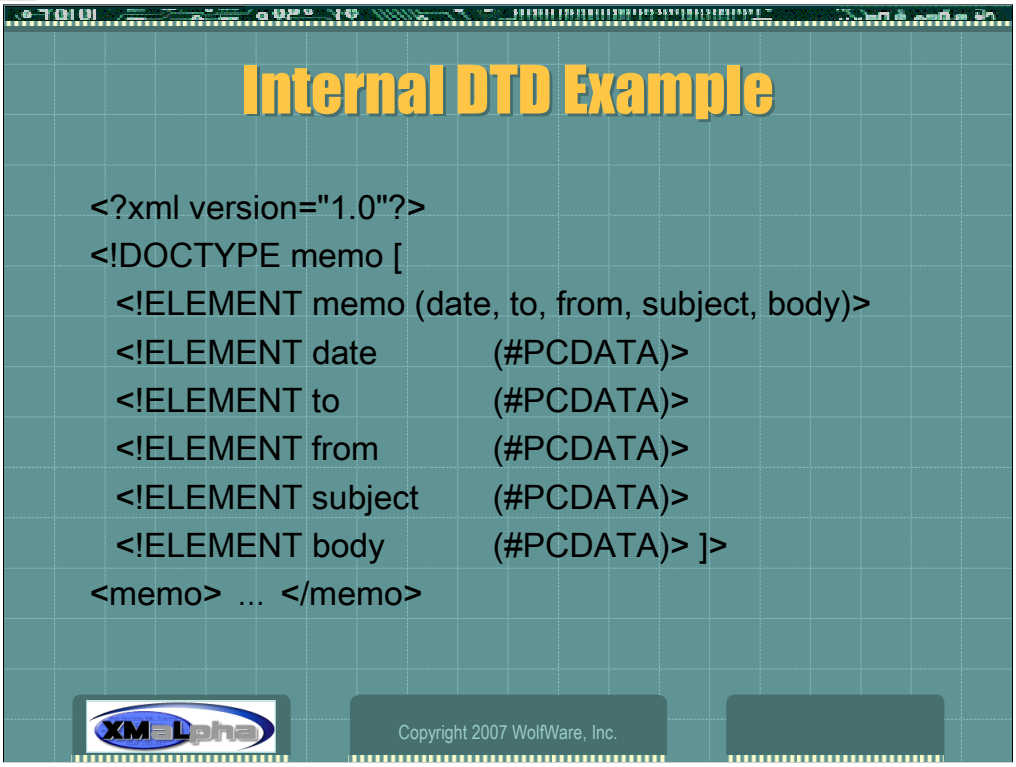
Does not allow as detailed of specification as Schema.

Internal declaration is legal, but not particularly useful, since it would only apply to that given document.

Again, this is analogous to specifying the class definition that determines what instances are legal.







Notice where the DOCTYPE ends – the ']>'

Notice that the DOCTYPE is embedded inside the XML document (internal DTD)

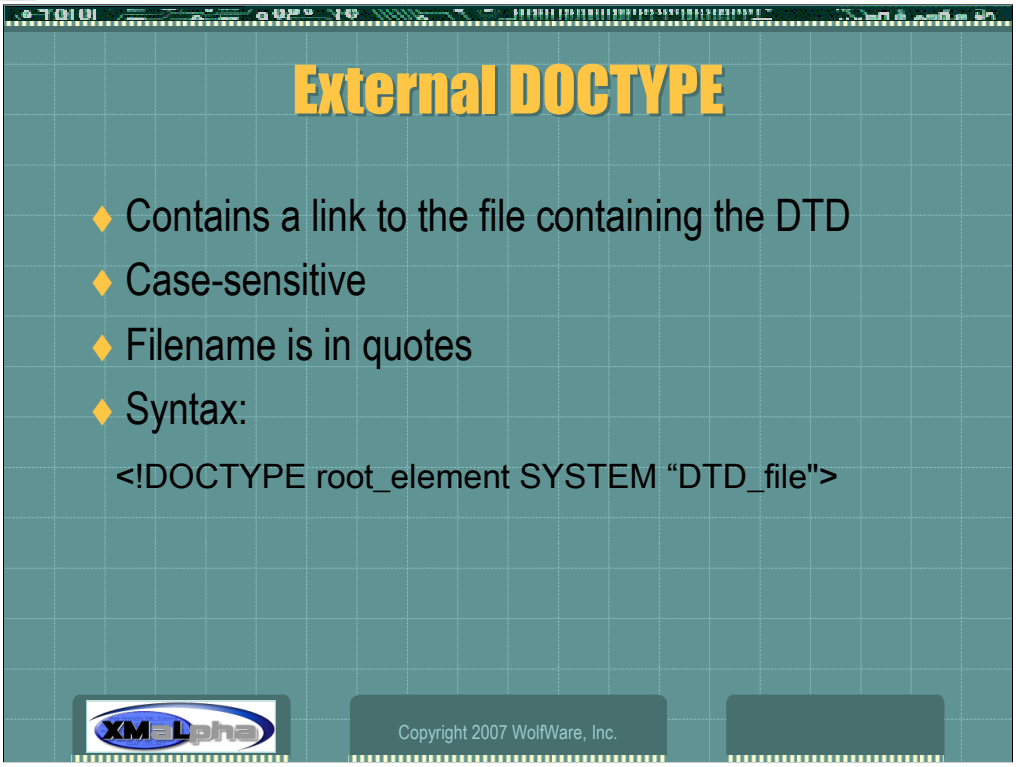
The XML memo follows the DOCTYPE statement

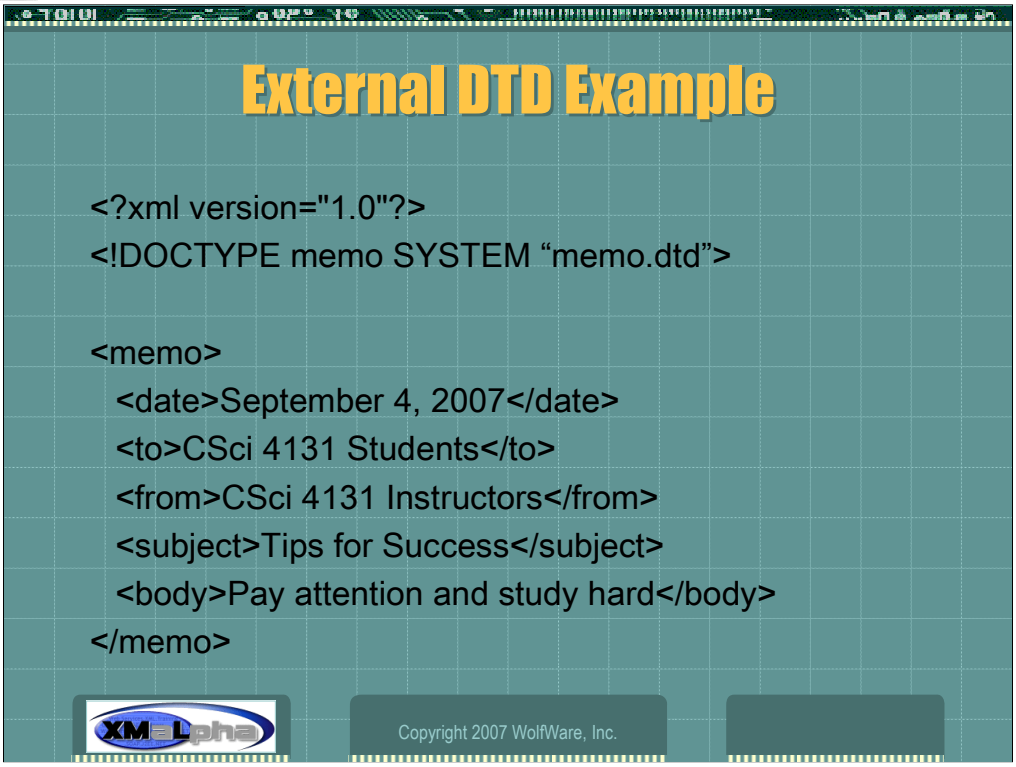
The first !ELEMENT is the same as the DOCTYPE (the root element)

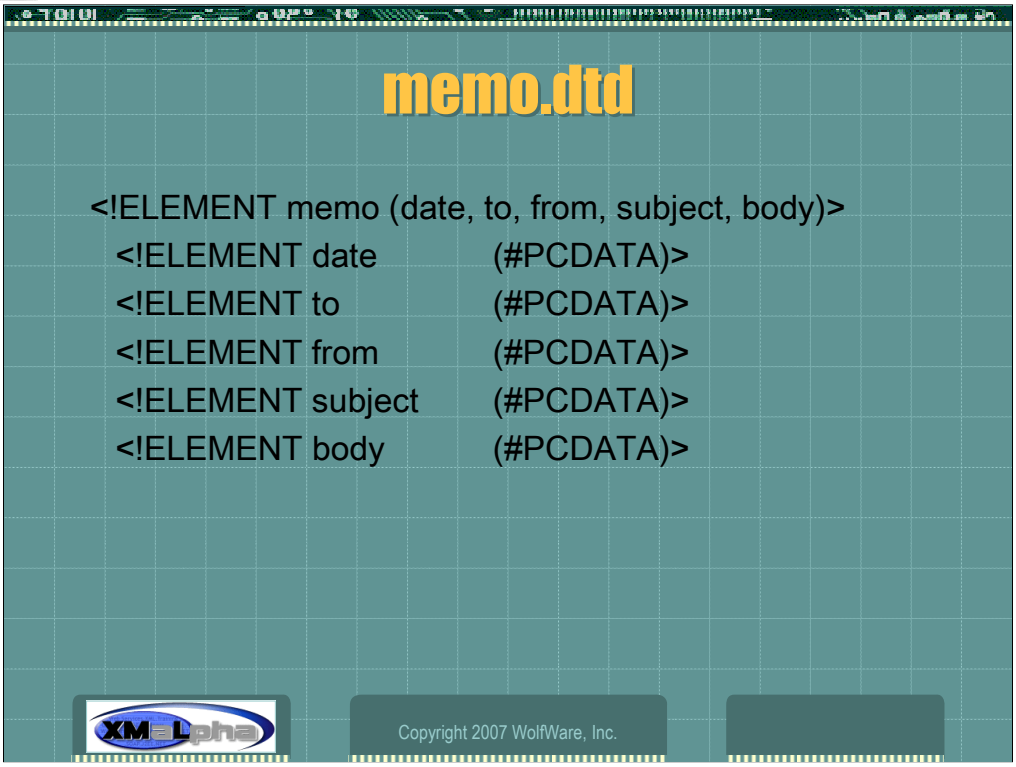
In this case, it says that the memo element contains 5 children elements: date, to, from, subject, body

The rest of the !ELEMENT definitions say that each contains text

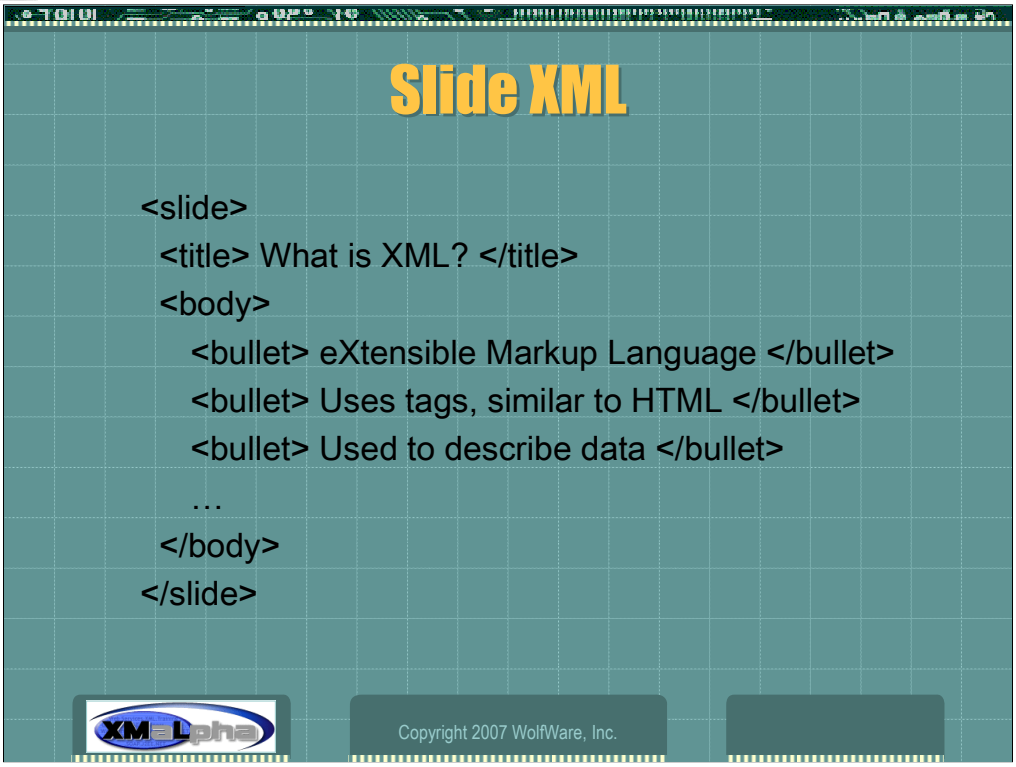
We'll see more about PCDATA later.



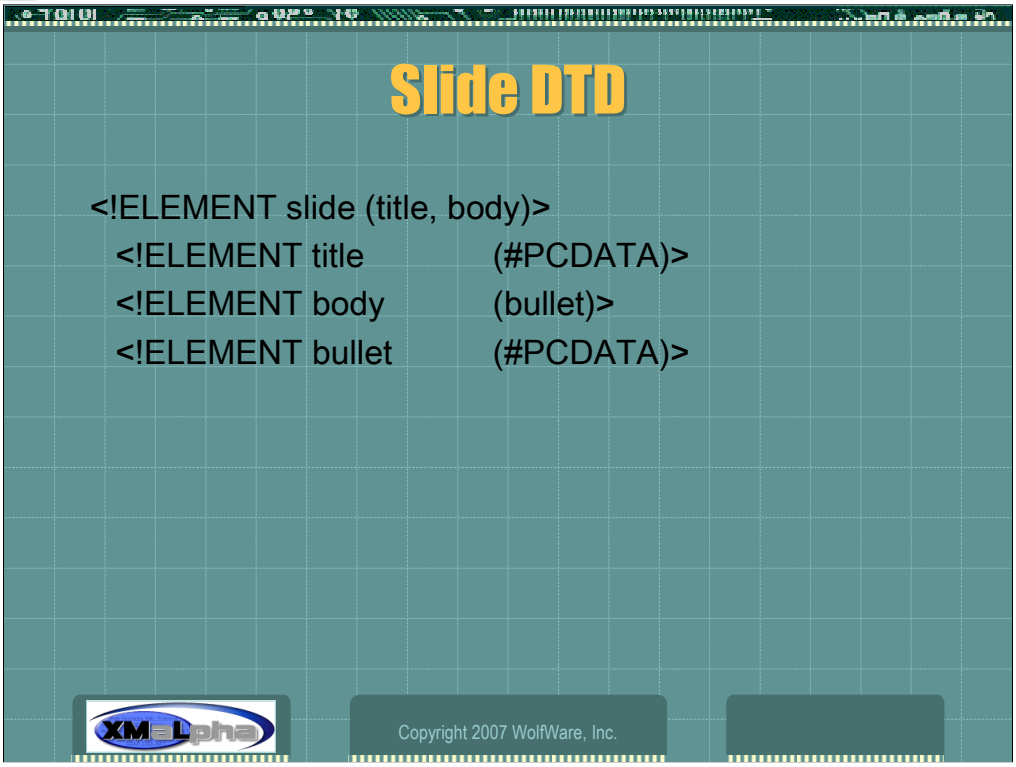




We'll still see more about PCDATA later.



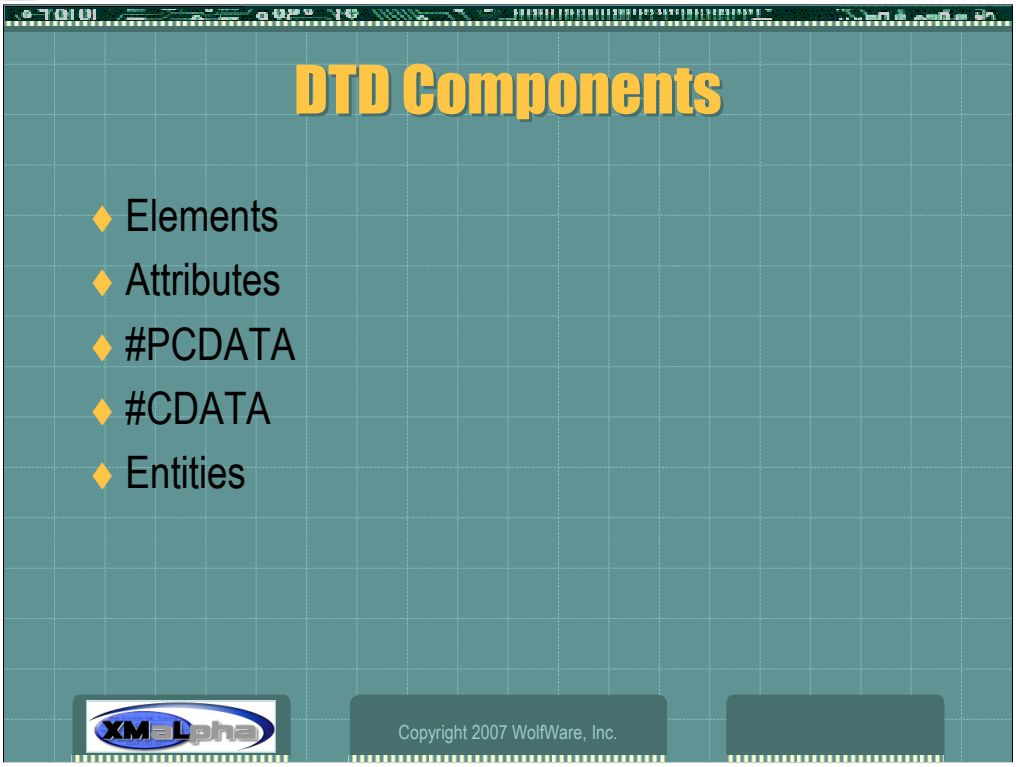
Recall slide from beginning of class



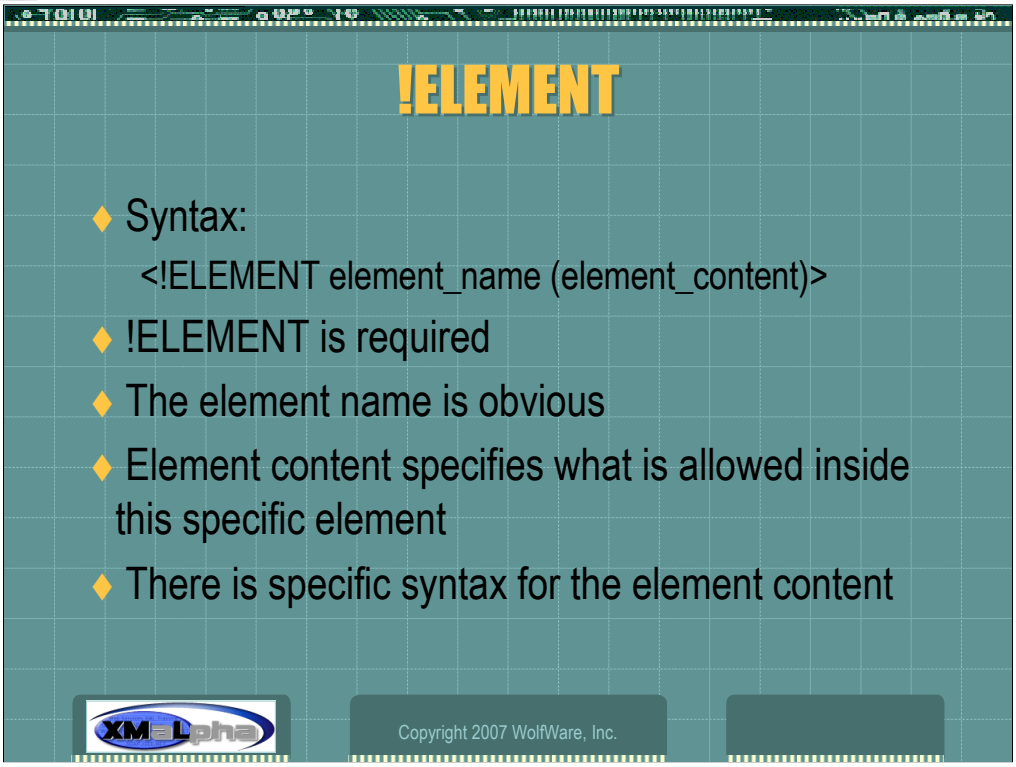
Is this sufficient? NO

It shows that body contains a child element, but only one.

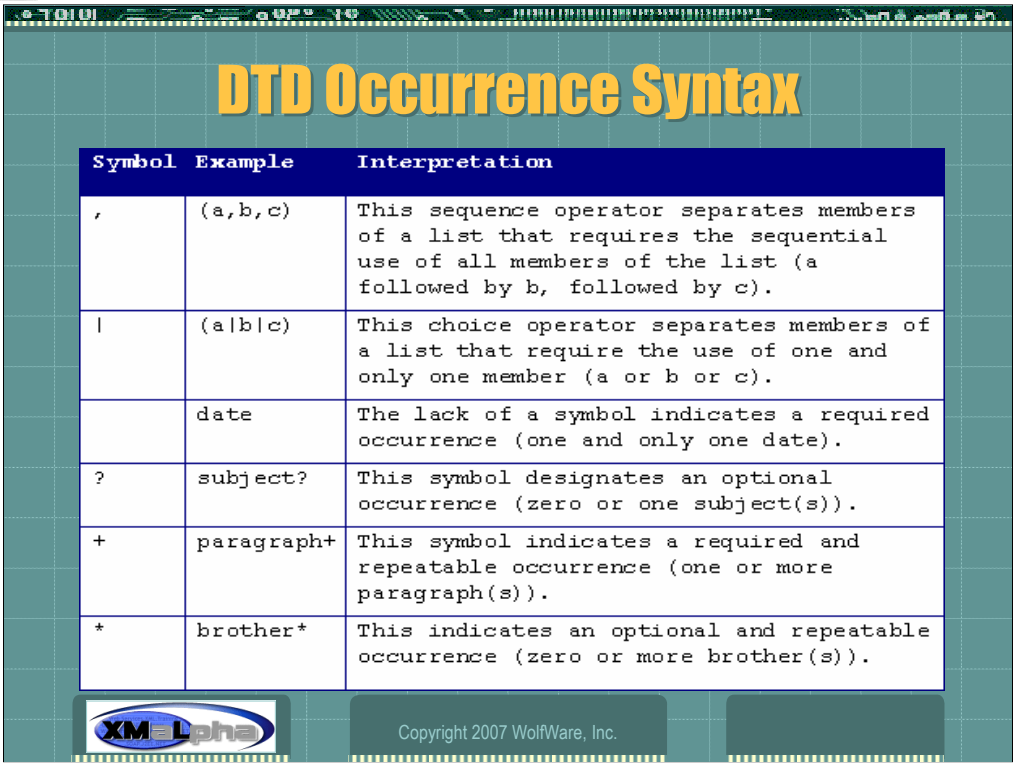
How do we specify more than one?



We'll look at each of these in the following slides.



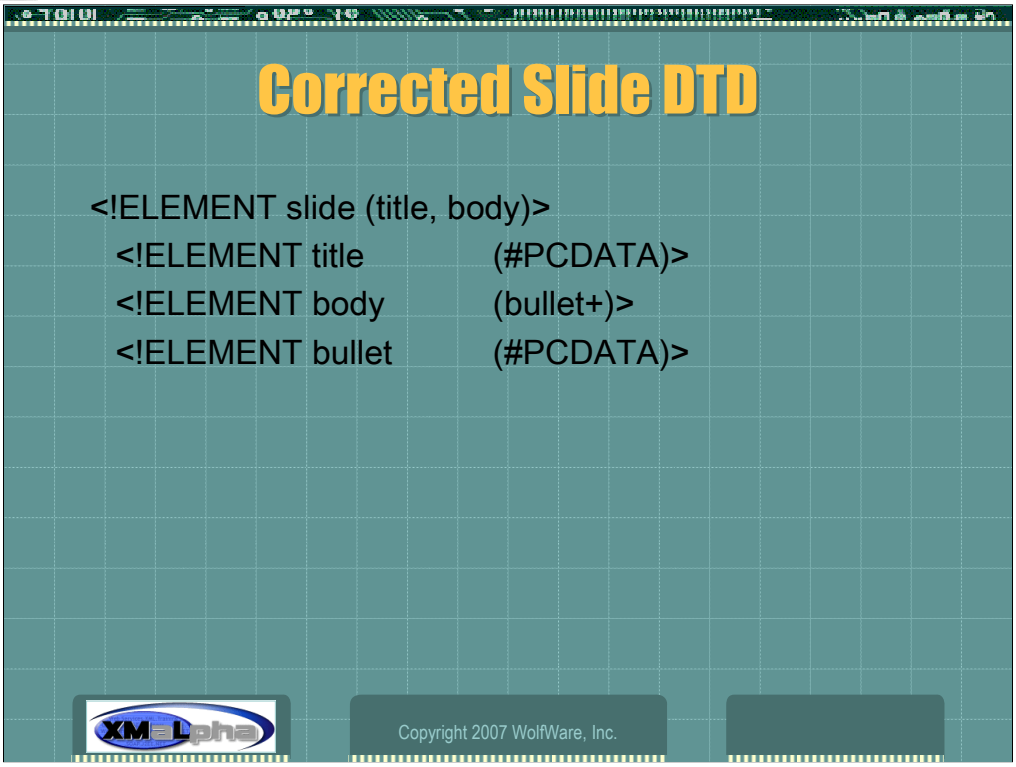
These rules are on the next slide.



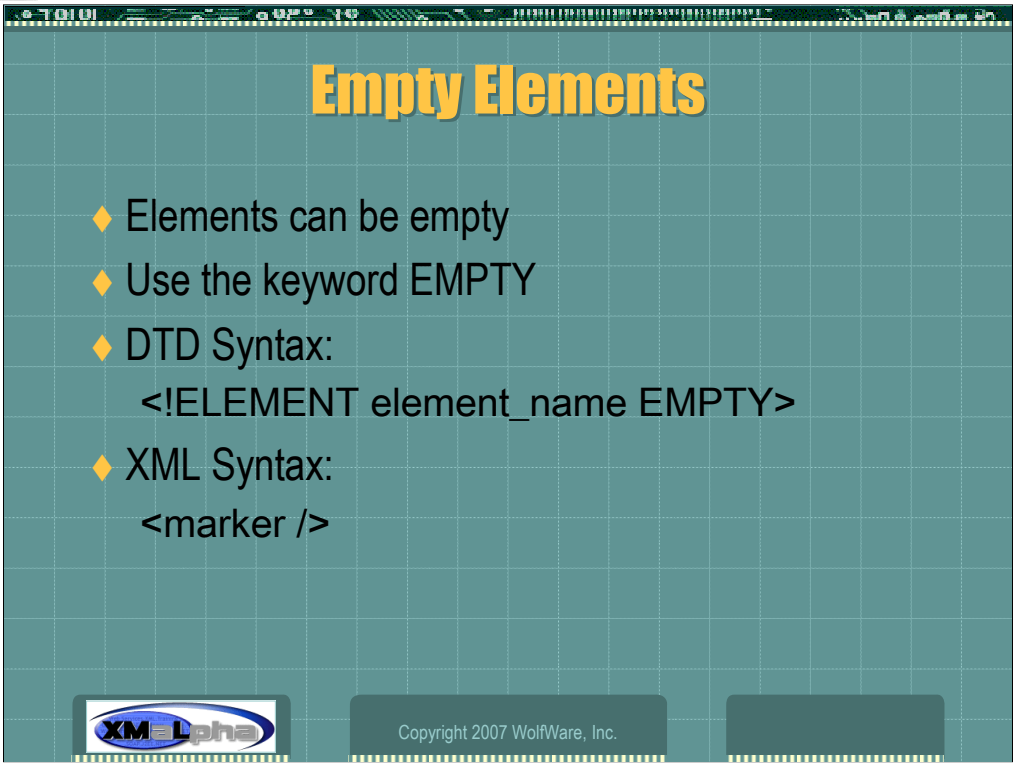
Give credit: Slide is from XMaLpha, Inc.

Should give some examples and see if students can read them.

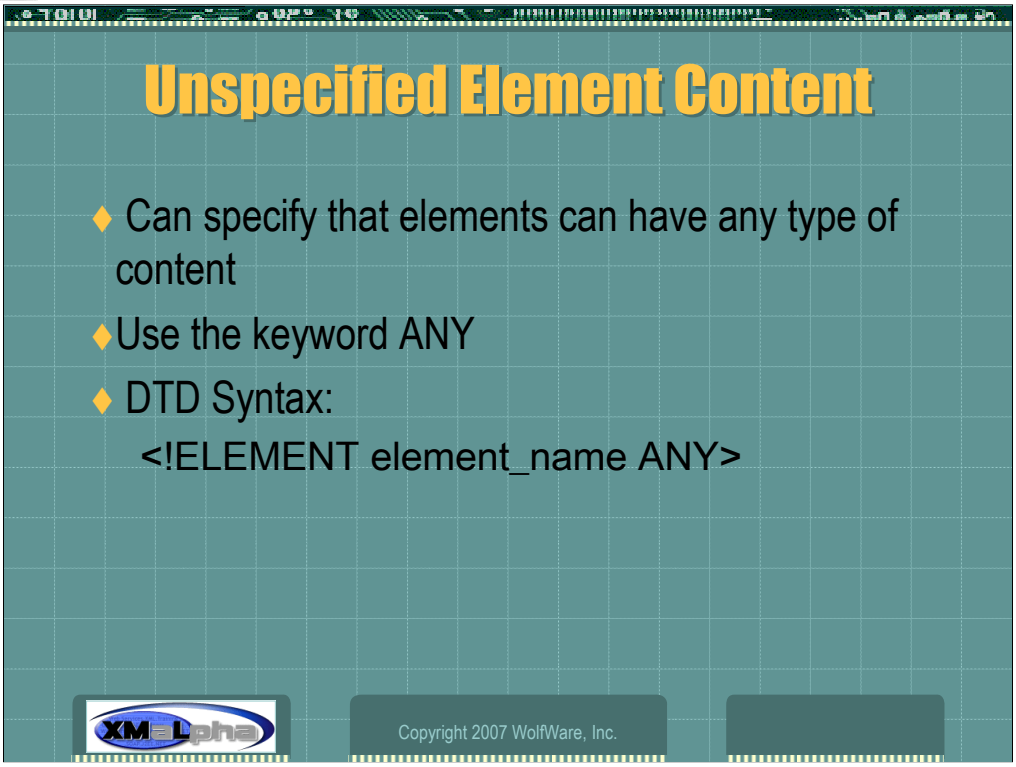
Then ask students to create some examples.

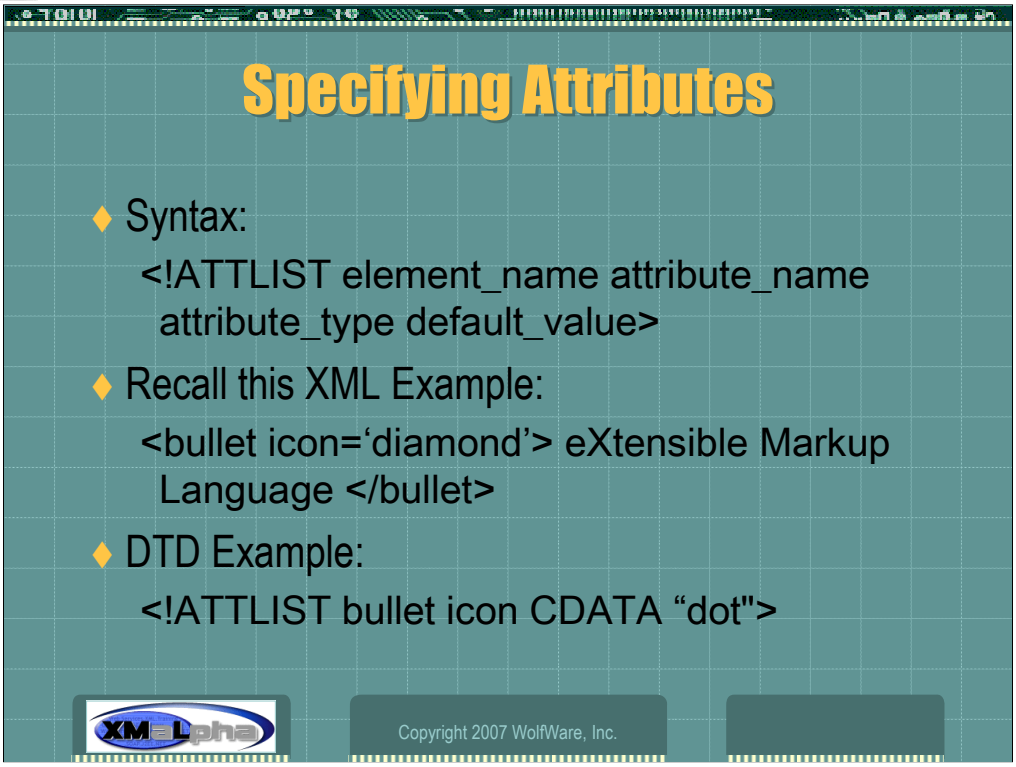


Assuming you are going to require at least one bullet.



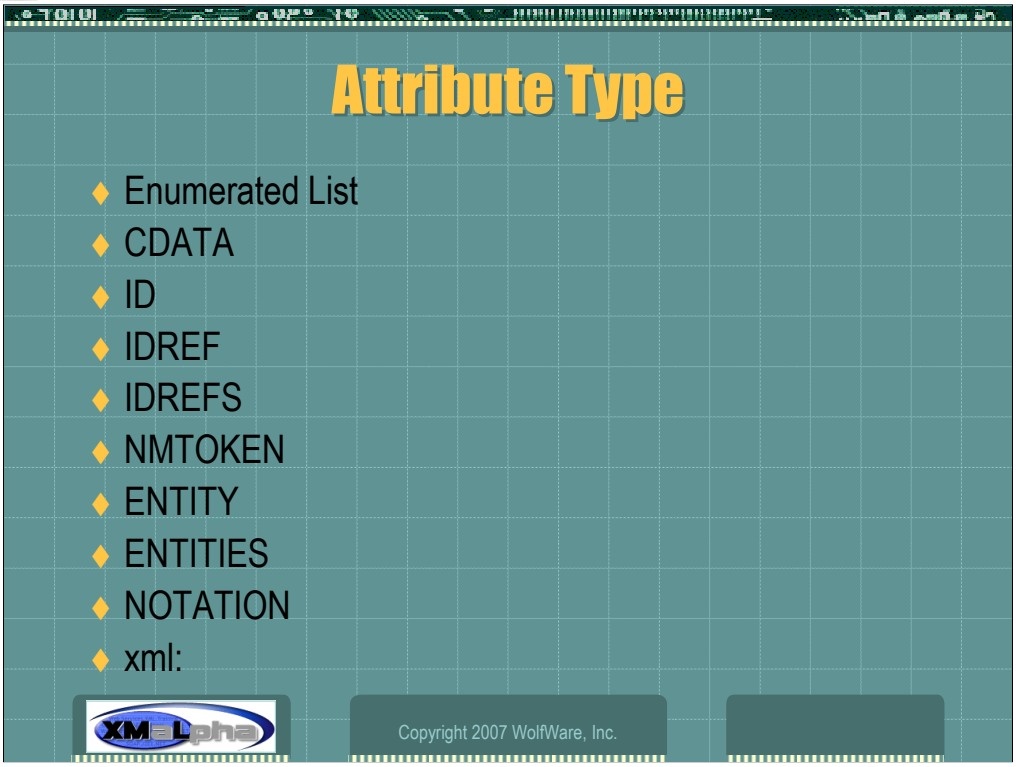
An HTML example is br.



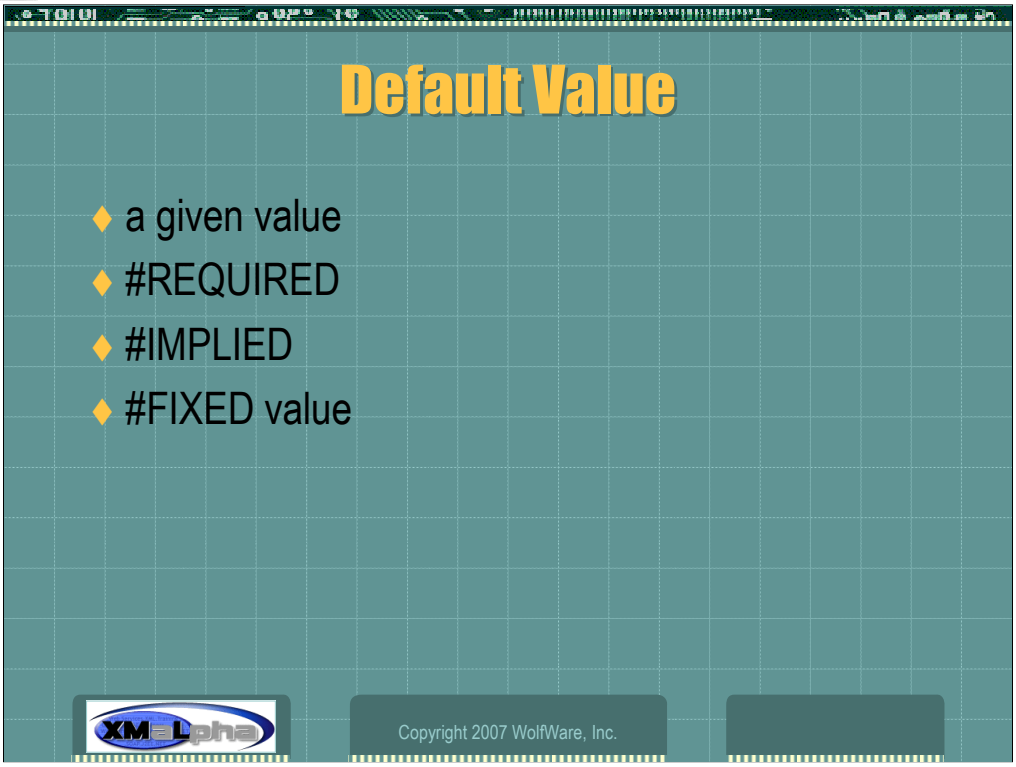


This says that the bullet element can have an icon attribute which is CDATA, and if it's not specified, the value is dot.

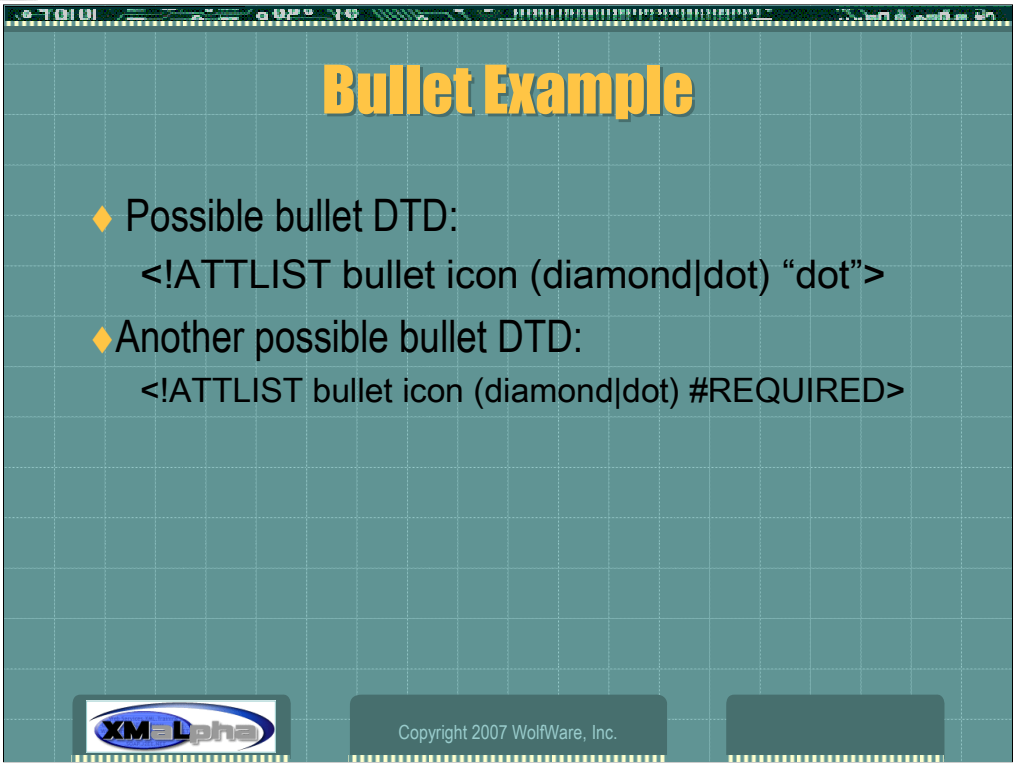
But we need to look at more than this.



We're only concerned with the first 2 in this class. The rest are shown for completeness.



We're only concerned with the first 2 in this class. The rest are shown for completeness.

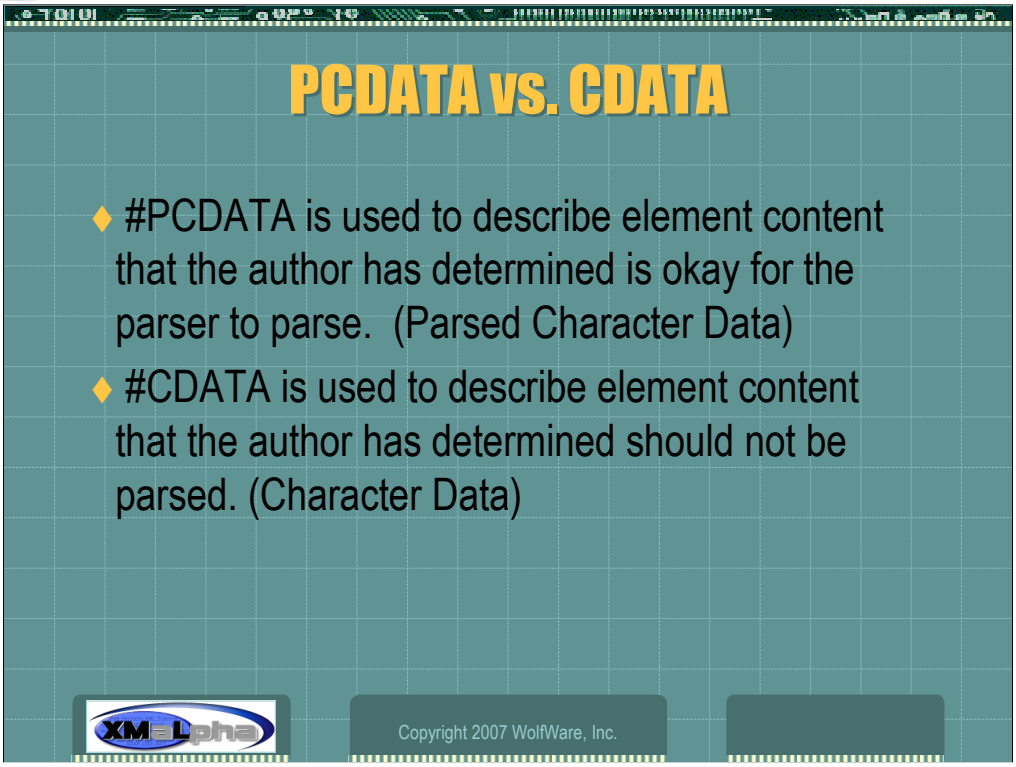


The first one says that the bullet element can have an icon attribute with a value of either diamond or dot and if not specified, the value is dot.

The second one says that the bullet element must have an icon attribute with a value of either diamond or dot.

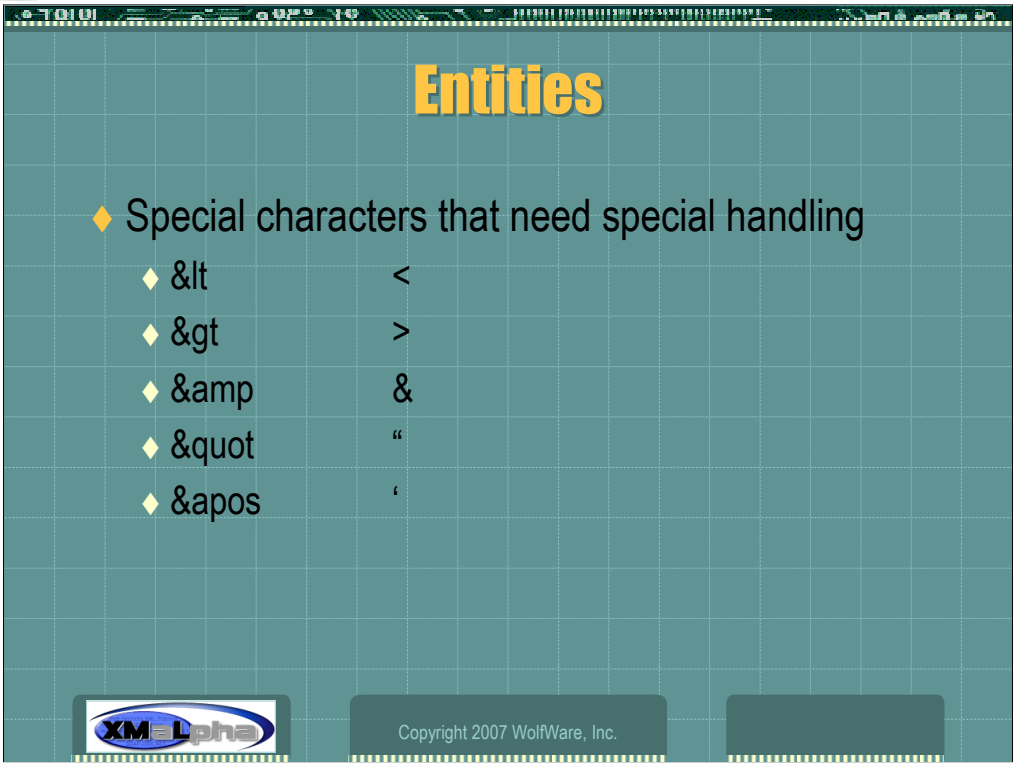
Should give some examples and see if students can read them.

Then ask students to create some examples.



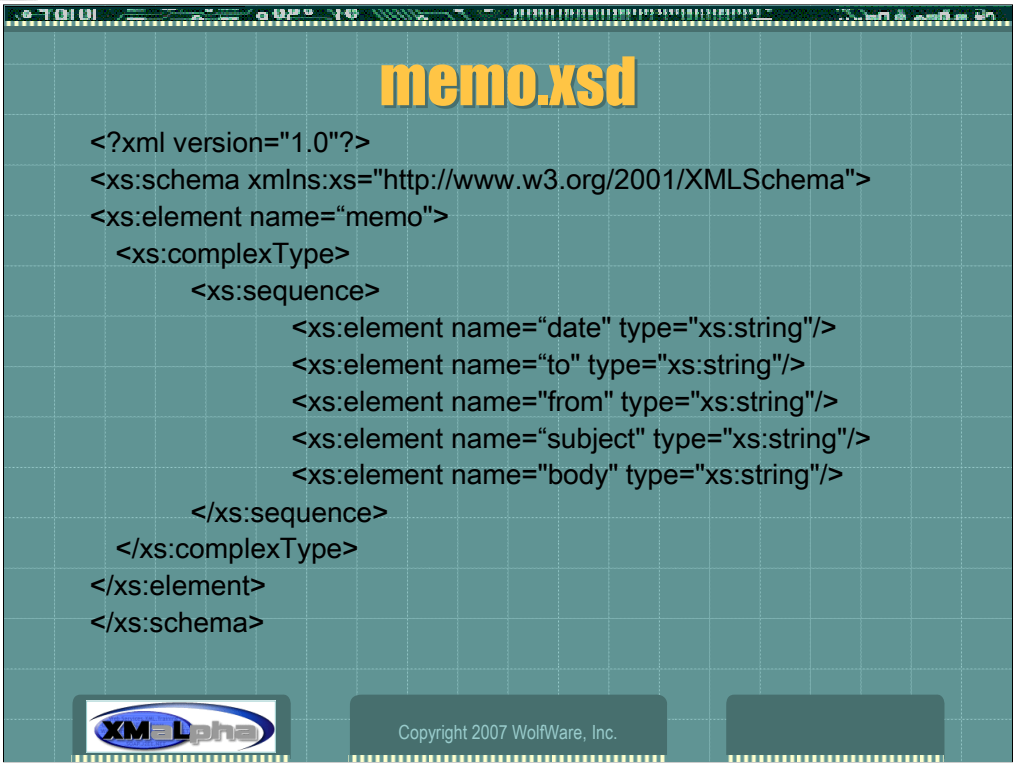
Talk about parsers and why you would/wouldn't want element content parsed.

This leads into the entities topic on the next slide.



These characters have special meaning in XML, so if you want to use them in PCDATA, you need to handle them specially.

Discuss why data types are important



This is a simple version, as it does not use namespaces.

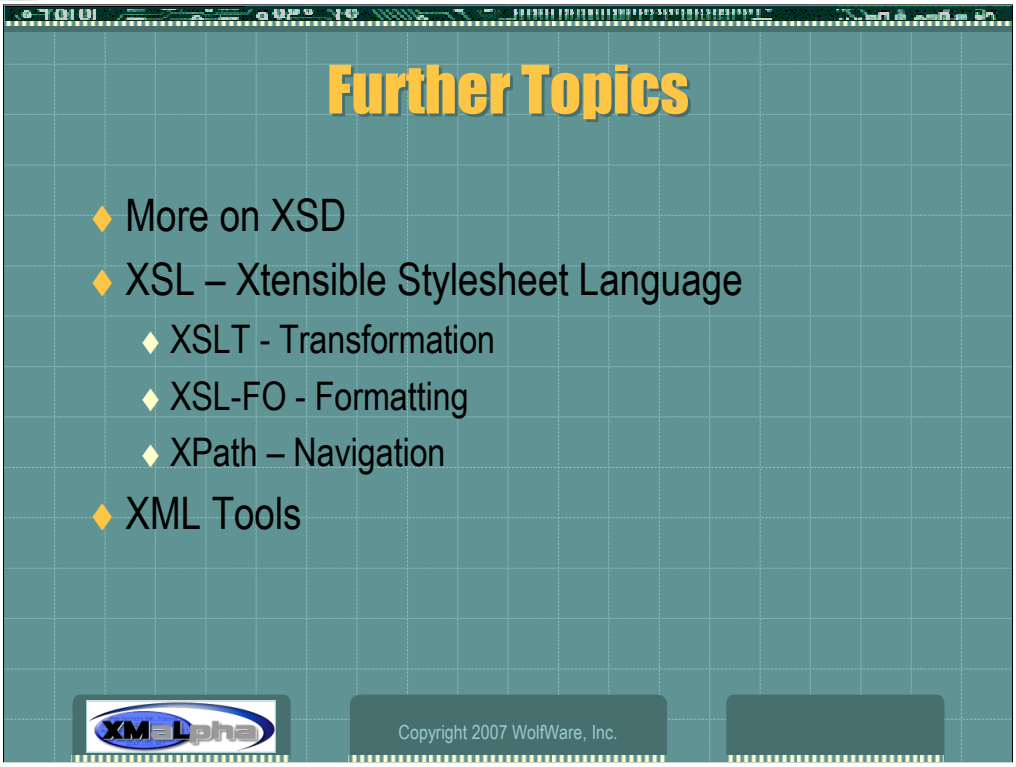
The first line says this is an XML document.

The second line ties this xml file to the w3 xml schema definition. This will ensure that this document follows the rules of XSDs.

Memo is a complex type because it contains other elements. It also specifies what namespace this document uses (xs). Notice that the other nodes are prefaced with this namespace.

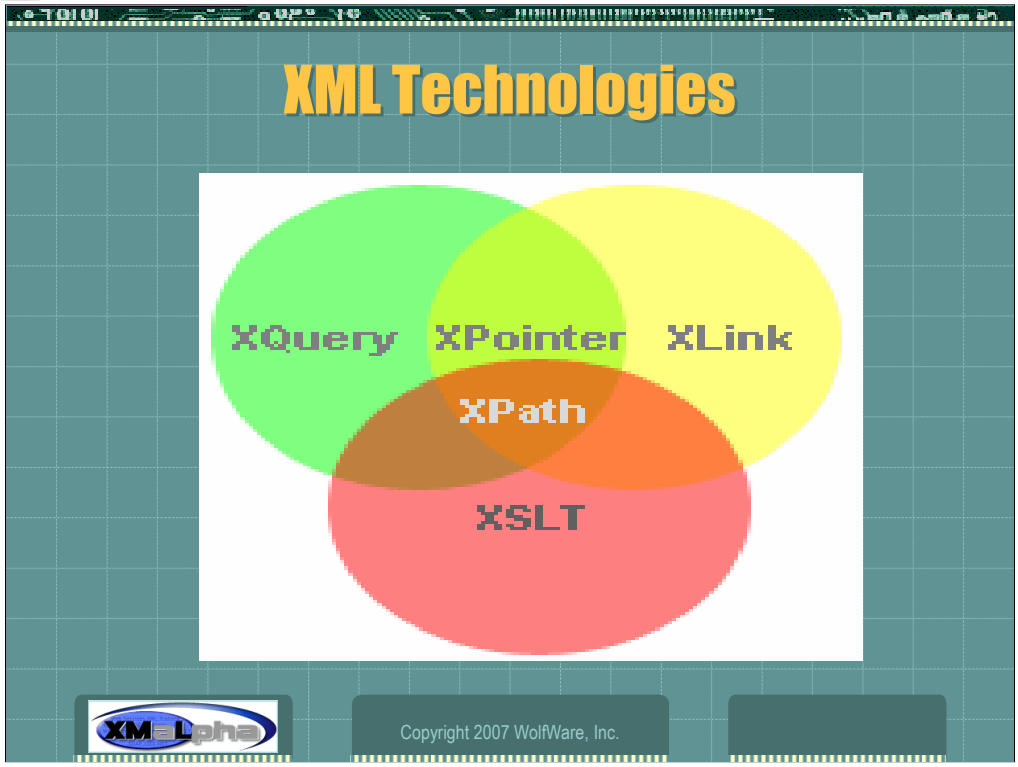
GTC – Namespaces are beyond the scope of this course.

UMN – We'll talk about namespaces later.

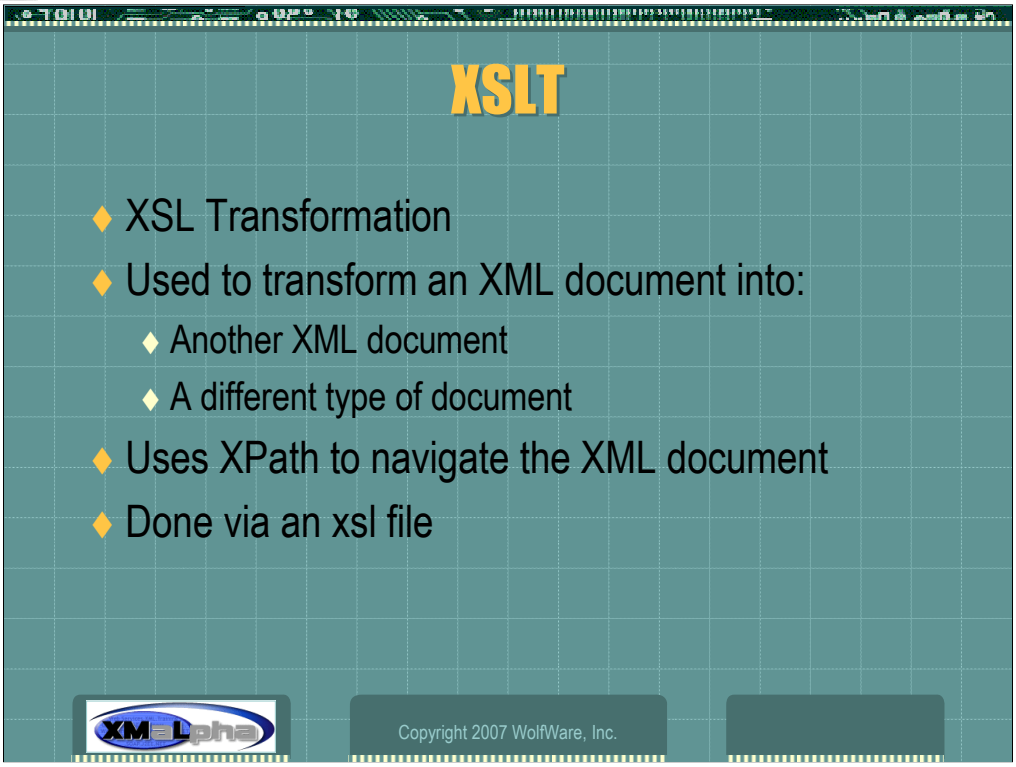


We've only briefly looked at XSD. There is a lot more to learn there.

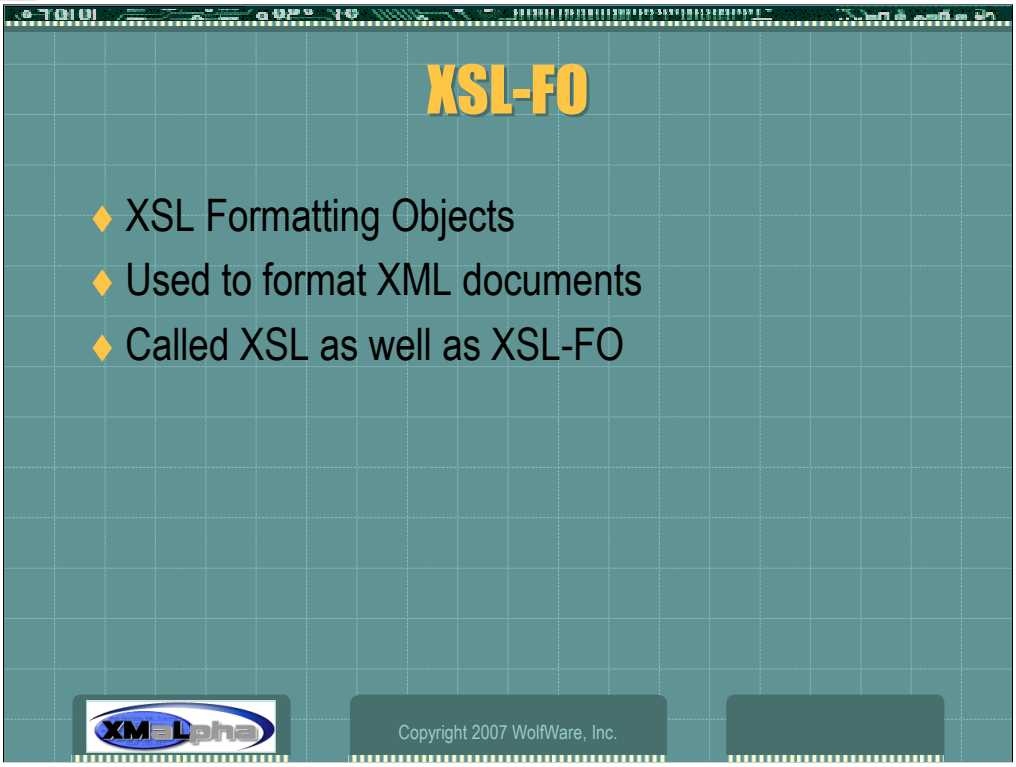
The next step would be XSL, which is a family of technologies



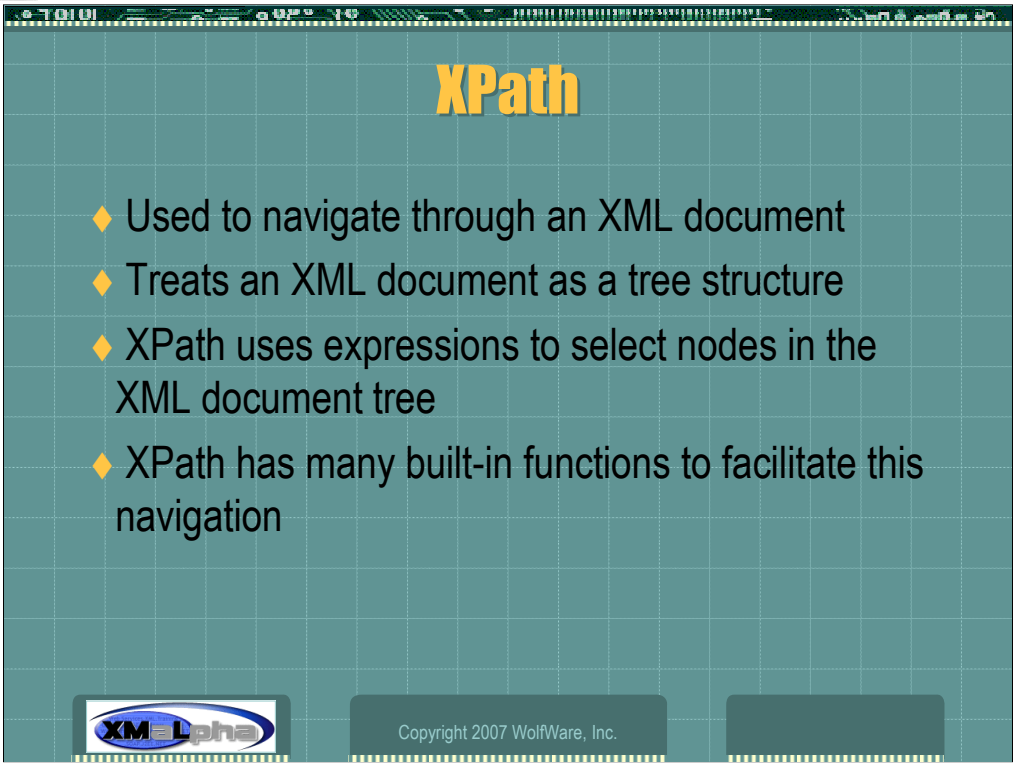
Credit: source is w3schools.com.



View and go through a sample xsl file.

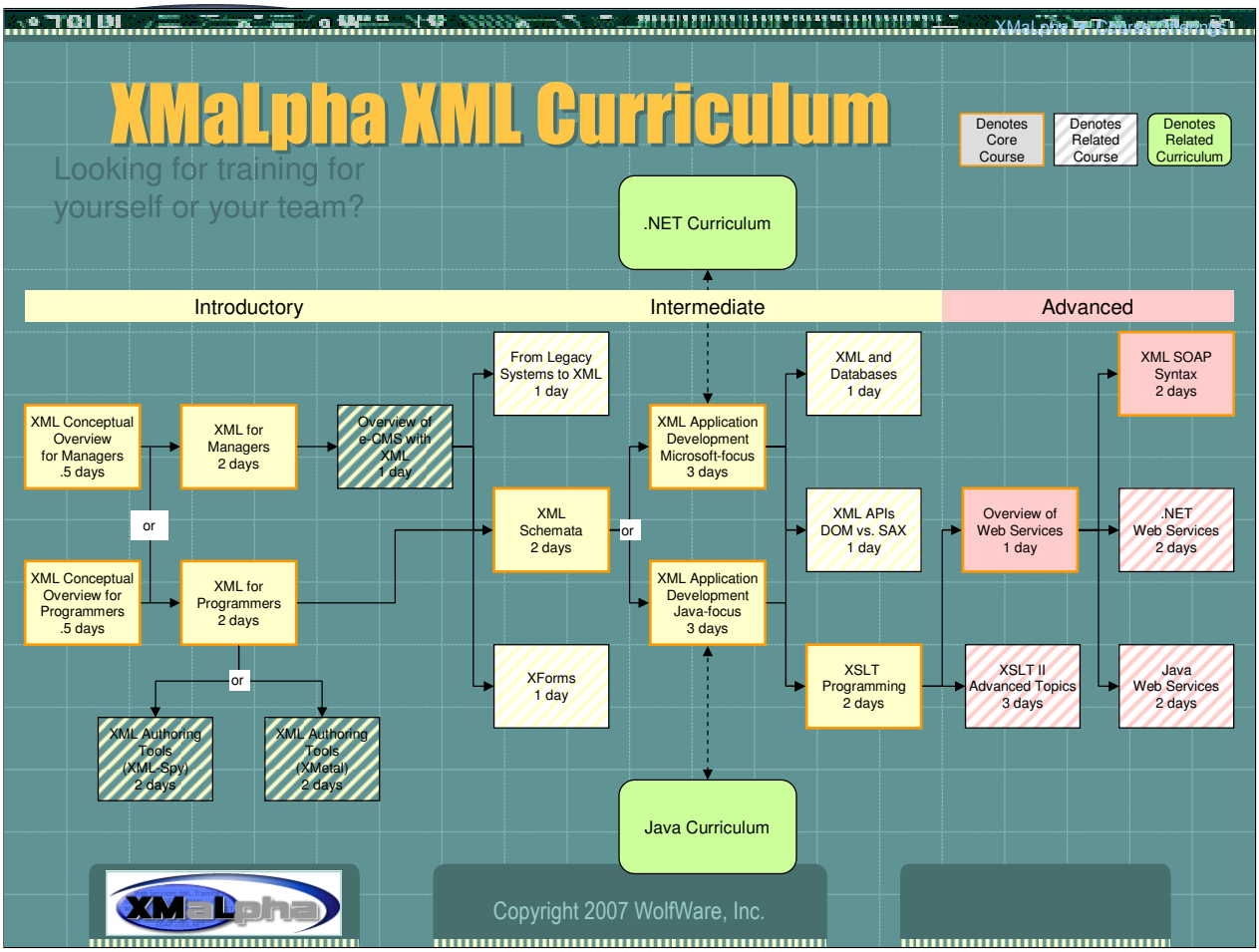


View and go through a sample xsl file.

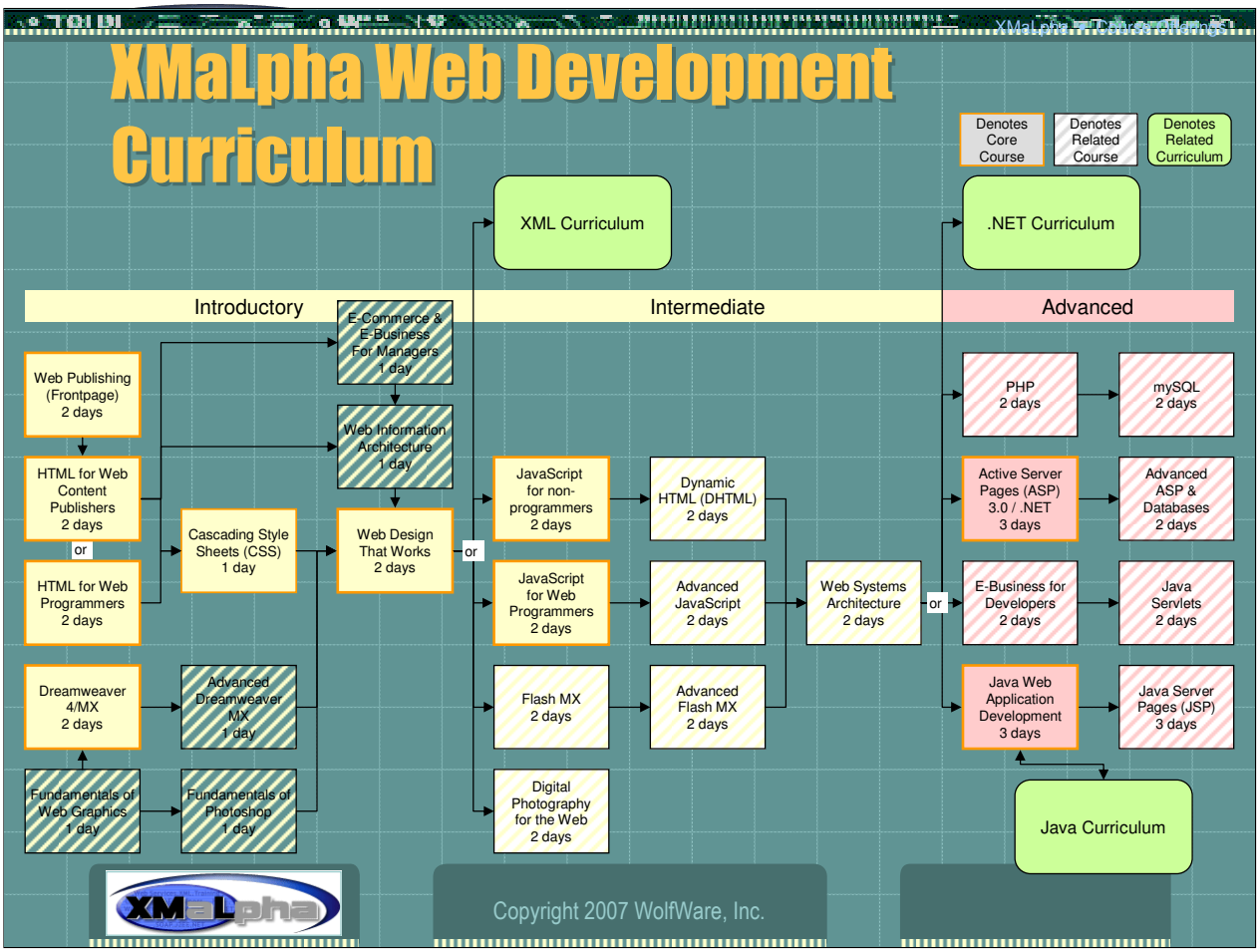


Go through some Xpath examples.

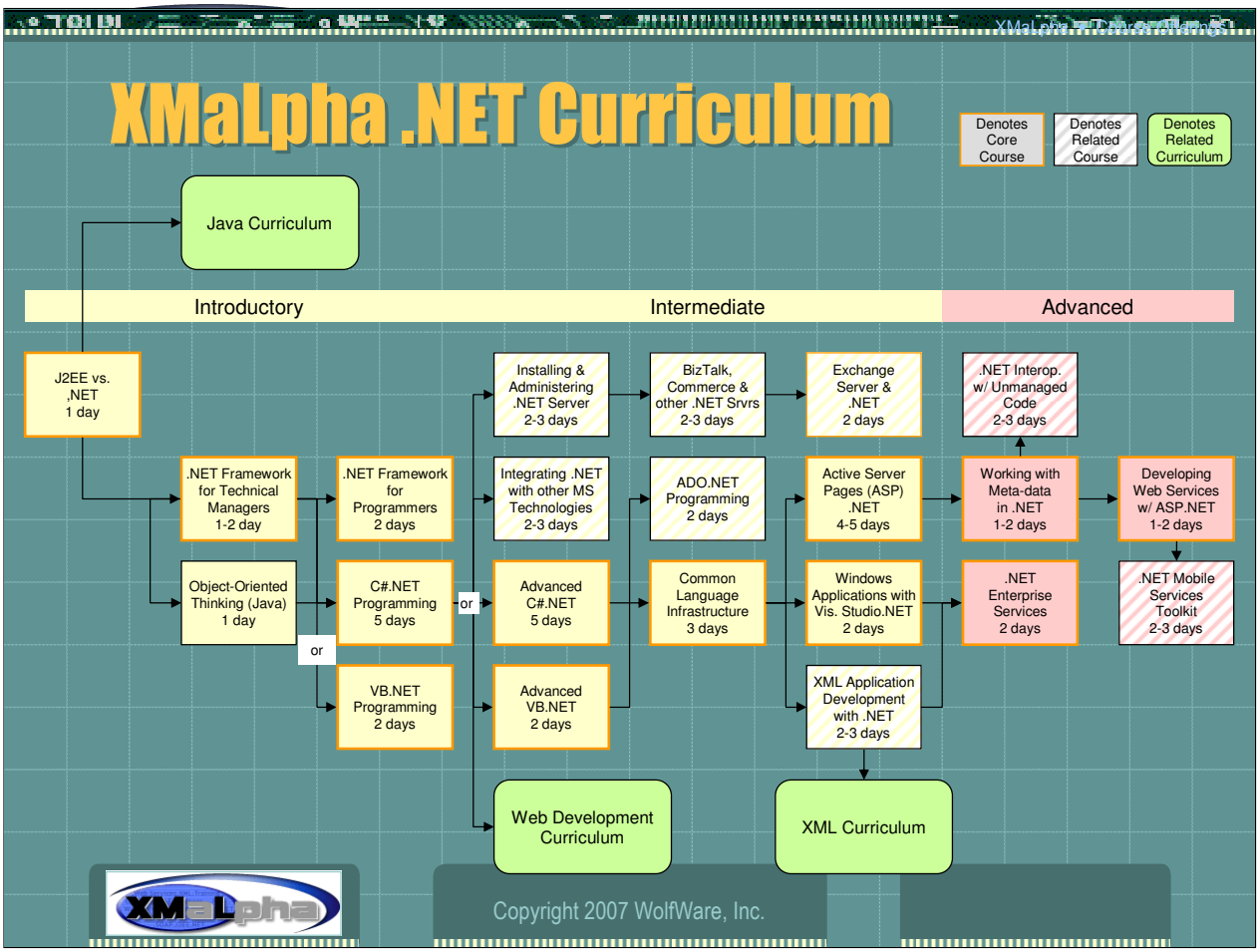
Show XMLSpy.



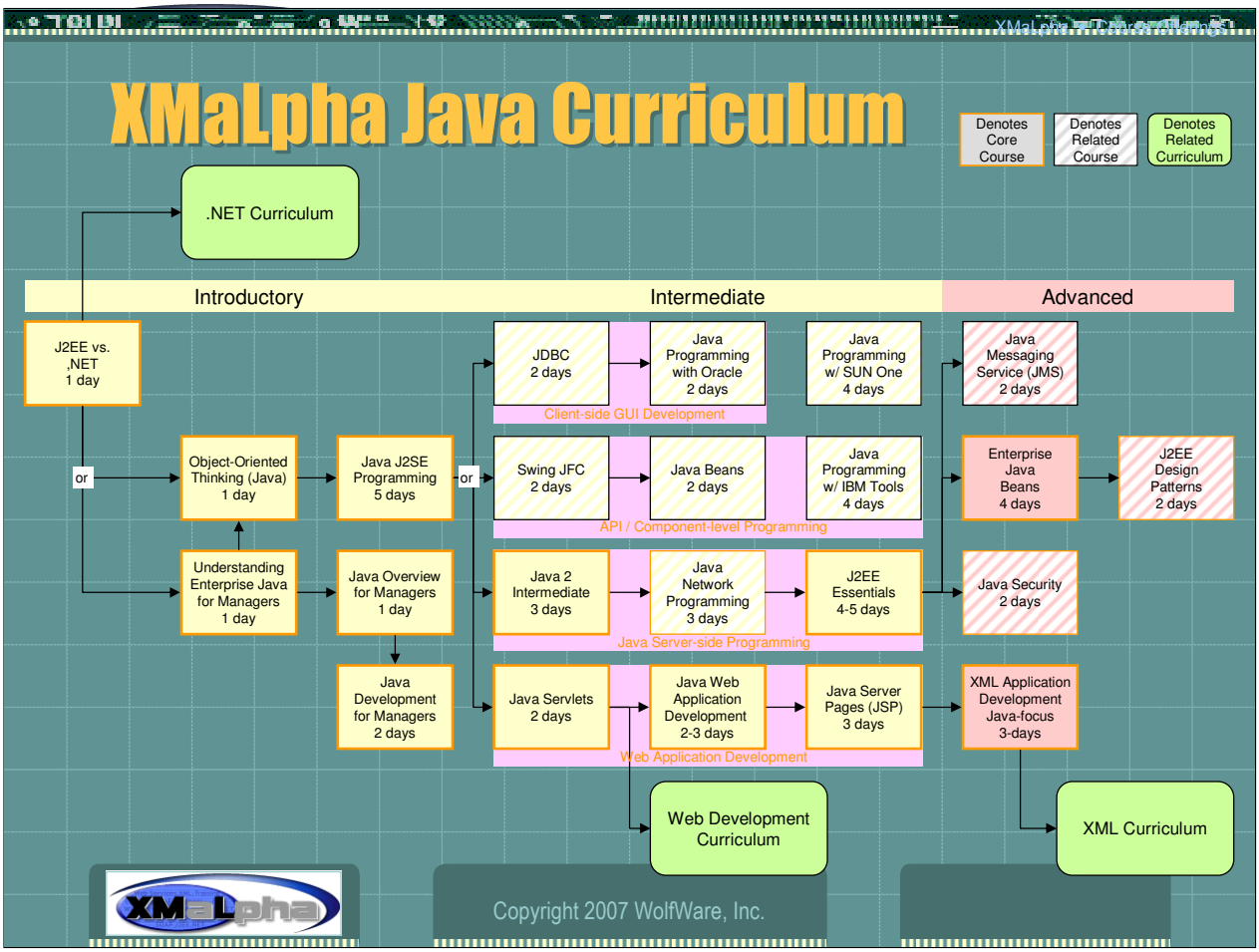
A place for your notes…



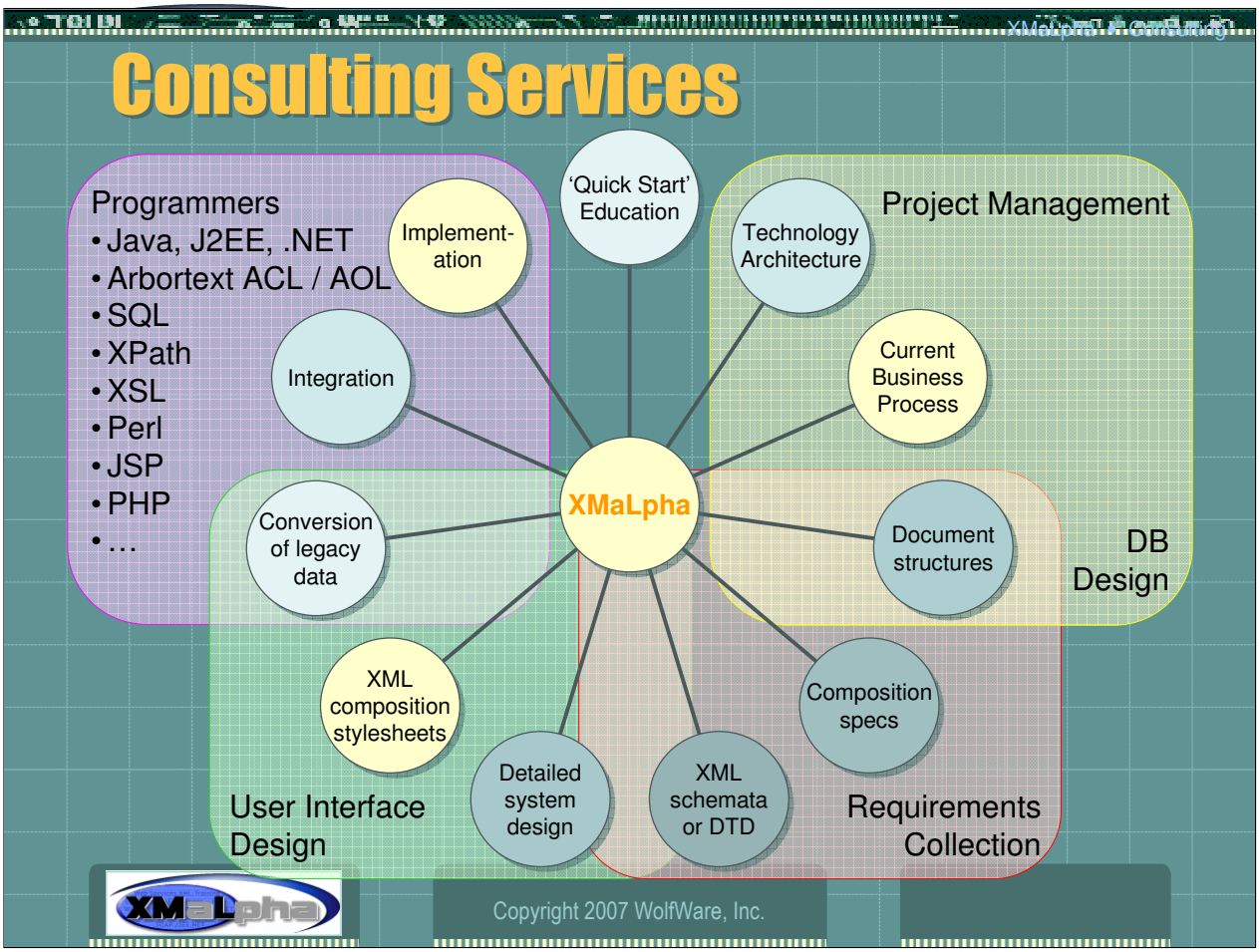
A place for your notes…



A place for your notes…



A place for your notes…



An independent perspective

A proven track record with extensive XML experience in the legislative environment

XMaLpha Technologies:

- Focus on design and analysis using our experience plus proven techniques
- Build architectural roadmaps and specifications
- Develop the core services using the selected tools and environments
- Design, plan, and deliver integration and implementation