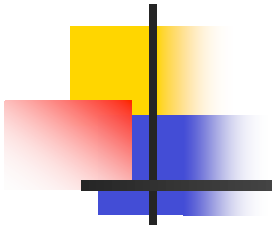
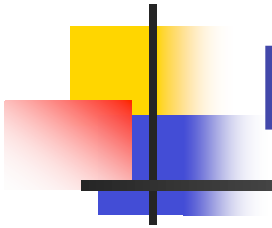


B336 Internet Systems Programming

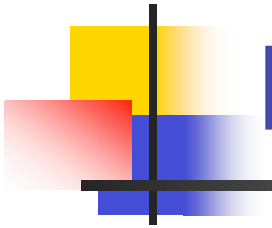
The Document Type Definition

(Week 5 Lecture 2-2)



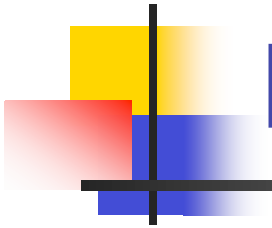
Lecture Objectives

- Understand what a DTD is, and its role in an XML document.
- Know the format and syntax of DTDs as specified by W3C's XML 1.0 Recommendations.



Learning Objectives

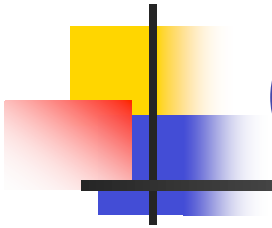
- In the scheme of what we are doing in this unit:
  - We are studying how to use XML as an important set of Internet technologies to use as solutions in different areas.
  - The ability to create new mark-up languages is a key factor in XML technologies.
  - The DTD is technically one of the main methods used to define the syntax of an XML document, and therefore it is the mechanism used to specify a new mark-up language.



Lecture Outline

- What is a DTD, and a *valid* XML document?
- The syntax of DTDs.
- DTDs and Validating Parsers.

The Document Type Definition (DTD)



- In the last lecture, we discussed how W3C's XML 1.0 Recommendations specify the syntax of a well-formed XML document.

- Part of the syntax allows a **Document Type Definition** to be included in the Prolog.

Prolog

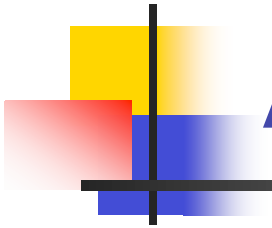
```
<?xml version="1.0"?>
<?xml stylesheet type="text/xsl" href="ss.xsl"?>
<!DOCTYPE course [
      <!ELEMENT course (name, unit+)>
      <!ELEMENT name (#PCDATA)>
      <!ELEMENT unit (title, lecturer*, tutor*)>
      <!ELEMENT title (#PCDATA)>
      <!ELEMENT lecturer (surname, othernames?, email*)>
      <!ELEMENT tutor (surname, othernames?, email*)>
      <!ELEMENT surname (#PCDATA) >
      <!ATTLIST surname language CDATA "English">
      <!ELEMENT othernames (#PCDATA) >
      <!ATTLIST othernames language CDATA "English">
      <!ELEMENT email (#PCDATA) >
]>

<course>
     <name>Bachelor of Science - Internet Computing</name>
     <duration>3 years</duration>
     <unit>
          <title>B336 Internet Systems Programming</title>
          <lecturer>
              <surname language="English">Hiew</surname>
              <othernames language="English">Hong Liang</othernames>
              <email>h.hiew@murdoch.edu.au</email>
          </lecturer>
     </unit>

     ...

</course>
```

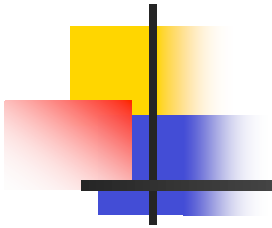
DTD



A Valid XML Document

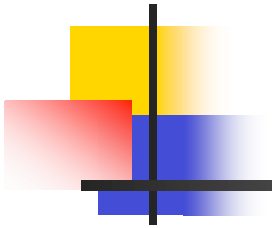
- The DTD defines a structure for the tags in the document.
  - What tags are allowed.
  - Which tag contains which tag.
  - Where are the basic text data.
  - etc.
- The XML 1.0 Recommendations defines a **valid** document to be one which
  - has a DTD, and
  - the document syntax conforms to the rules of its own DTD.

Creating New Mark-up Languages



- Since DTDs defines the tags and how they relate to each other, they basically define a new mark-up languages.

- It is up to groups, organizations, consortiums, partners, etc, to
  - decide on the mark-ups that are most appropriate for their area
  - put down the structure of the tags in DTD,
  - and then all their shared documents can make use of the same DTD.



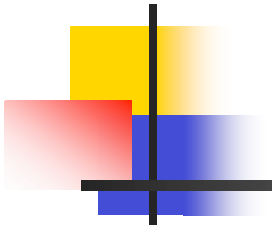
Declaration for a DTD

- DTDs can be put into the XML documents themselves (see example again).

- This DTD section of an XML document is called the Document Type ***Declaration***.

  - The declaration is in the form of a `<!DOCTYPE>` tag.
  - Internal DTDs are declared using the format

    `<!DOCTYPE` *`rootname`* `[` *`DTD`* `]>`



Defining Elements

- To declare the syntax for elements, we use the tag of the form

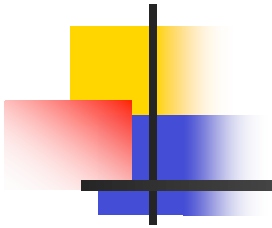
```
<!ELEMENT name content_model>
```

  - where

    `name` is the name of the element/tag

    `content_model` is the one of

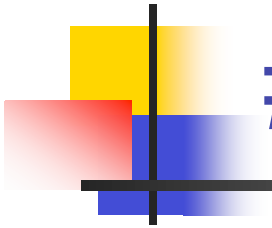
    - `EMPTY` (meaning an empty element)
    - `ANY` (meaning the parser should not check the content)
    - mixed (meaning #PCDATA or elements), or
    - child elements



Defining Elements

- Some examples:

```
<!ELEMENT book_header (title, author)>
<!ELEMENT staff (name, duty) >
<!ELEMENT document EMPTY >
<!ELEMENT undefined ANY >
<!ELEMENT info (#PCDATA) >
<!ELEMENT info ((field, value)| #PCDATA) >
```

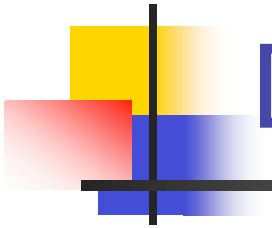


#PCDATA

- To define a section that contains the raw text data, we use the keyword `#PCDATA` (for **parsed character data**).

- The `#PCDATA` is still parsed, so cannot contain characters like “<“.

  - This is different from the CDATA section mentioned in the last lecture, which is not parsed.

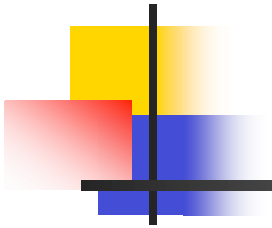


Defining Children

- To define that one element contains another element, we use the name of the element in the *content_model*.

- Eg.

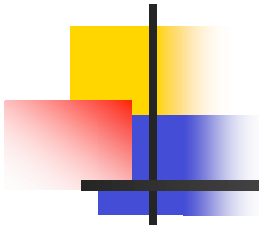
```
<!ELEMENT course (name, unit+)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT unit (title, lecturer*, tutor*)>
...
```



Having Multiple Children

- To define that one element contains more than one child, we use special symbols:

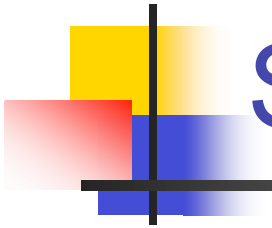
x+	one or more occurrences of x
x*	zero or more occurrences of x
x?	zero or one occurrence of x
x, y	x followed by y (ie. a **sequence**)
x | y	x or y, but not both.



Having Multiple Children

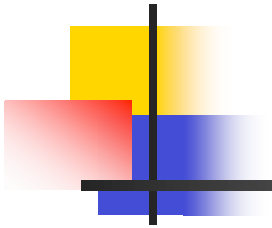
- Some examples:

```
<!ELEMENT book_header (title?, author?, info*)>
<!ELEMENT staff (name, duty+) >
<!ELEMENT info ((field?, value)| #PCDATA) >
```



Subsequences with parentheses

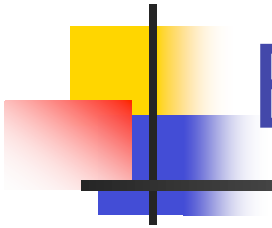
- You can include subsequences of child elements by using parentheses in the child sequence.



DTD Comments

- Comments can be put anywhere in the DTD section, using the same comment syntax as defined by a well-formed XML document.
- Eg.

```
<!DOCTYPE course [
      ...
      <!-- This the name of the course -->
      <!ELEMENT name (#PCDATA) >
      ...
]>
```

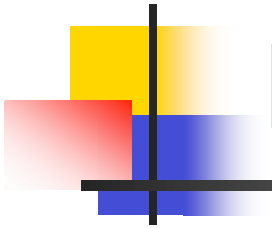


External DTDs

- DTDs which are not defined in the XML document itself can be referred by using a different declaration.

- Eg.

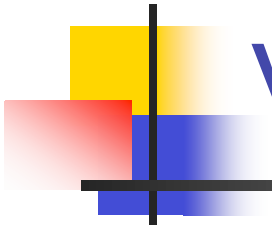
```
<!DOCTYPE poetry SYSTEM "course.dtd">
<!DOCTYPE poetry PUBLIC
  "-//uni consortium//Custom Course v1.0//EN”
  "http://university.edu.au/course.dtd”>
```



Multiple DTDs

- You can define an XML using elements from multiple DTDs.

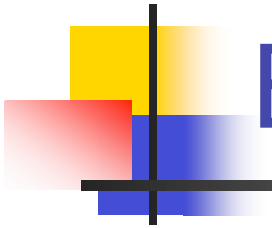
- This is done using the concept of Namespaces, which we will cover in future lectures.



Validating Parsers

- All XML parsers require that the documents be well-formed, but some do not check for validity.

- Parsers which validates the documents they parse against the document's DTD are called **validating parsers**.



Extra Reading

- Required Reading:
  - Chapter 6 in the Sybex textbook - for details of creating DTDs.
- Official Reference
  - XML 1.0 Recommendation
    - http://www.w3.org/TR/REC-xml