



ĐẠI HỌC BÁCH KHOA HÀ NỘI  
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

# XML, DTD và XML Schema

# Nội dung

1. Giới thiệu và cú pháp XML
2. Đặc tả nội dung với DTD
3. Đặc tả nội dung với XMLSchema

# 1. Giới thiệu và cú pháp XML

1. Giới thiệu XML
2. Đặc điểm XML
3. Ứng dụng XML
4. Cú pháp XML

Định chuẩn của XML

Thẻ khai báo tham số

Thẻ chỉ thị xử lý

Thẻ ghi chú

Thẻ CDATA và PCDATA

5. Cấu trúc, đặc tả cấu trúc và nội dung của XML
6. Sử dụng thẻ thực thể, tên thẻ
7. Namespace

# Giới thiệu XML

- XML: eXtensible Markup Language - là một ngôn ngữ đánh dấu được sử dụng để tạo ra thẻ riêng, tạo nên các văn bản với dữ liệu tự mô tả.
- Được tạo nên bởi Liên minh mạng toàn cầu WWW (W3C) nhằm khắc phục những hạn chế của HTML - ngôn ngữ đánh dấu siêu văn bản. Giống như HTML, XML cũng được dựa trên SGML – Standard Generalized Markup Language.
- Là cơ sở của nền công nghệ thương mại điện tử, các công ty đang sử dụng XML để giải quyết những vấn đề kinh doanh.

# Giới thiệu XML

- Văn bản có cấu trúc XML cho phép **biểu diễn thông tin về các đối tượng trong thực tế**
- XML dùng để phục vụ cho việc mô tả dữ liệu (*thông tin lưu trữ bao gồm những gì, lưu trữ ra sao*) để các hệ thống khác nhau có thể đọc và sử dụng những thông tin này một cách thuận tiện
- Các thẻ (*tag*) của XML thường không được định nghĩa trước mà chúng được tạo ra **theo quy ước** của người, (hoặc *Chương trình*) tạo ra XML theo những quy ước riêng.
- XML sử dụng các khai báo kiểu dữ liệu **DTD** (Document Type Definition) hay lược đồ **Schema** để **mô tả dữ liệu**.

# Giới thiệu XML

HTML	XML
HTML được thiết kế cho mục đích <b>trình bày dữ liệu</b>	XML được thiết kế cho mục đích <b>lưu trữ và truyền tải dữ liệu</b> giữa các hệ thống khác nhau
HTML dùng để hiển thị dữ liệu và chú trọng vào việc <b>dữ liệu được hiển thị như</b> thế nào	XML dùng để mô tả dữ liệu và chú trọng vào <b>nội dung của dữ liệu</b>
HTML hiển thị thông tin	XML mô tả thông tin

# Ưu điểm XML

- Dễ dàng đọc và phân tích dữ liệu, do XML dùng **text** để mô tả dữ liệu nên các chương trình đều có thể đọc được XML dễ dàng
- Dễ dàng tạo 1 file XML.
- Lưu trữ cấu hình cho web site

# Đặc điểm của XML

- XML cung cấp một phương tiện dùng văn bản (*text*) để mô tả thông tin và áp dụng một **cấu trúc kiểu cây** cho thông tin đó.
- Tại mức căn bản, mọi thông tin đều thể hiện dưới dạng text, chen giữa là các **thẻ đánh dấu (*markup*)** với nhiệm vụ ký hiệu sự phân chia thông tin thành một cấu trúc có thứ bậc, các **phần tử (*element*)** dùng để chứa dữ liệu và các **thuộc tính** của các phần tử đó.



# Ứng dụng của XML

Ví dụ tệp **web.xml**:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>

<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
  version="2.4">

  <display-name>HelloWorld Application</display-name>
  <description>
    This is a simple web application with a source code organization
    based on the recommendations of the Application Developer's Guide.
  </description>

  <servlet>
    <servlet-name>HelloServlet</servlet-name>
    <servlet-class>examples.Hello</servlet-class>
  </servlet>

  <servlet-mapping>
    <servlet-name>HelloServlet</servlet-name>
    <url-pattern>/hello</url-pattern>
  </servlet-mapping>

</web-app>
```

# Ứng dụng của XML

## Ví dụ tệp RSS:

```
<?xml version="1.0" encoding="UTF-8" ?>
<rss version="2.0">

<channel>
  <title>W3Schools Home Page</title>
  <link>https://www.w3schools.com</link>
  <description>Free web building tutorials</description>
  <item>
    <title>RSS Tutorial</title>
    <link>https://www.w3schools.com/xml/xml_rss.asp</link>
    <description>New RSS tutorial on W3Schools</description>
  </item>
  <item>
    <title>XML Tutorial</title>
    <link>https://www.w3schools.com/xml</link>
    <description>New XML tutorial on W3Schools</description>
  </item>
</channel>

</rss>
```

# Cú pháp XML

- Văn bản có cấu trúc XML cho phép biểu diễn thông tin về các **đối tượng trong thực tế**.
- Đối tượng  $x$  thuộc loại  $X$  trong thực tế được biểu diễn bởi **thẻ  $X$**  trong tài liệu XML bao gồm cả các **thuộc tính  $a$**  của  $x$ .
- Ví dụ:

Phân số 4/5 trong thực tế có thể:

`<PHAN_SO Tu_so="4" Mau_so="5" />`

# Cú pháp XML

- Ví dụ (tiếp):

Dãy các số nguyên a bao gồm các số nguyên 1,4,5,-3 sẽ được biểu diễn bởi thẻ

```
<DAY_SO>
```

```
<SO Gia_tri="1" />
```

```
<SO Gia_tri="4" />
```

```
<SO Gia_tri="5" />
```

```
<SO Gia_tri="-3" />
```

```
</DAY_SO>
```

# Định chuẩn XML

- **Hệ thống các thẻ** đánh dấu:
- Các thẻ đánh dấu trong ngôn ngữ theo định chuẩn XML bao gồm 2 loại: Thẻ có nội dung và thẻ rỗng.
  - Các thẻ có **nội dung** có dạng:  
`<Tên> Nội dung </Tên>`
  - Các thẻ rỗng có dạng:  
`<Tên />`
- Các thẻ có thể có hoặc không có các thuộc tính (trong cùng thẻ ). **Thuộc tính** trong một thẻ có dạng:  
`Ten_thuoc_tinh="Gia_tri"`

# Định chuẩn XML

- Ví dụ : Tài liệu XML

```
<?xml version="1.0" encoding="utf-8"?>  
<DUONG_TRON Ban_kinh="5">  
    <DIEM x="4" y="2"/>  
</DUONG_TRON>
```

# Định chuẩn XML

- **Quan hệ lồng nhau** giữa các thẻ có nội dung:
  - Nội dung bên trong thẻ có nội dung có thể là các thẻ khác. Khi thẻ A có nội dung là thẻ B ta gọi: Thẻ A là thẻ cha của B , thẻ A chứa thẻ B.
  - Qui định yêu cầu các thẻ với quan hệ lồng nhau hoàn toàn. Khi thẻ A là thẻ cha của thẻ B, A phải chứa phần bắt đầu và cả phần kết thúc của thẻ B.

# Định chuẩn XML

- Ví dụ:

Thẻ A là thẻ cha của B với dạng lồng nhau hoàn toàn (hợp lệ):

<A>

    <B> ....</B>

</A>

Thẻ A là thẻ cha của B với dạng lồng nhau không hoàn toàn (không hợp lệ):

<A>

    <B ></A>

</B>



# Định chuẩn XML

- Một tài liệu XML phải có duy nhất một và chỉ một thẻ chứa tất cả các thẻ còn lại, gọi là **thẻ gốc – Root element (Document element)**:
- Ví dụ:

```
<?xml version="1.0"?>
```

```
<Blog>
```

```
    <Title>Let me know what you think</Title>
```

```
    <Author>Yin Yang</Author>
```

```
</Blog>
```

# Định chuẩn XML

- **Well-formed** XML Document (đúng cú pháp) :
  - Có duy nhất có một phần tử thuộc cấp cao nhất trong tài liệu, còn gọi là nút gốc (*root element*).
  - Mỗi **một thẻ mở** đều **phải có thẻ đóng** và tên thẻ là phân biệt hoa thường.
  - Các thẻ khi đóng phải theo **đúng trình tự** (*mở sau đóng trước*)
  - **Tên thẻ** không nên có khoảng trắng, không nên bắt đầu bằng “xml”.
  - Các thuộc tính (*attributes*) của một thẻ luôn luôn tồn tại theo cặp theo quy ước: **<tên> = “<giá\_trị>”**; không nên đặt tên thuộc tính trùng nhau, và giá trị của thuộc tính phải đặt trong cặp dấu nháy kép hay nháy đơn. Tên của thuộc tính (*attribute*) sẽ theo qui luật đặt tên giống như đối với tên thẻ.
  - Các thẻ (tag) trong XML có thể **lồng nhau** (*Thẻ này có thể chứa nhiều thẻ khác ở bên trong*).

# Nội dung tài liệu XML

- Nội dung của tài liệu XML bao gồm 2 phần:
  - **Nội dung chính:** Hệ thống các thẻ đánh dấu (có hoặc không có nội dung) tương ứng với các **thông tin cần biểu diễn**.
  - **Nội dung phụ:** Hệ thống các thẻ khác có **ý nghĩa bổ sung, tăng cường một số thông tin** về tài liệu XML. Các thẻ này có tác dụng giúp cho việc sử dụng, xử lý trên tài liệu XML tốt hơn trong một số trường hợp nhất định.

# Nội dung tài liệu XML

- Các thẻ bên trong nội dung **phụ** bao gồm:
  - Thẻ khai báo tham số
  - Thẻ chỉ thị xử lý
  - Thẻ ghi chú
  - Thẻ CDATA
  - Thẻ khai báo cấu trúc (đặc tả cấu trúc với DTD)
  - Thẻ khai báo thực thể (Kĩ thuật đặc tả nội dung tài liệu XML)

# Thẻ khai báo tham số

- **Thẻ khai báo tham số**: mô tả thêm một số thông tin chung (tham số) về tài liệu XML ngoài các thông tin biểu diễn trong nội dung chính.
- Cú pháp:  
`<?xml Ten_1="Gia_tri_1" Ten_2="Gia_tri_2" ... ?>`
- Ten\_1, Ten\_2, ...: các tên các tham số và Gia\_tri\_1, Gia\_tri\_2, ... là các giá trị tương ứng. Có 3 tham số được dùng là version, encoding, và standalone.

# Thẻ khai báo tham số

- Tham số **version** : Khai báo về phiên bản của định chuẩn XML được sử dụng.

- Ví dụ :

Tài liệu XML thuộc định chuẩn 1.0

`<?xml version="1.0" ?>`

# Thẻ khai báo tham số

- Tham số **encoding** : Khai báo về cách mã hóa các ký tự trong tài liệu.
- Ví dụ: Tài liệu XML sử dụng cách mã hóa Unicode ký hiệu utf-8:

`<?xml version="1.0" encoding="utf-8" ?>`

Tài liệu XML sử dụng cách mã hóa Unicode ký hiệu utf-16:

`<?xml version="1.0" encoding="utf-16" ?>`

# Thẻ khai báo tham số

- Tham số **standalone** : Khai báo về liên kết tới tài liệu DTD bên ngoài
- Tham số này chỉ có 2 giá trị hợp lệ là “yes” , “no”. Giá trị định sẵn là “no”.
- Ví dụ: Tài liệu XML có liên kết với tài liệu DTD bên ngoài khác:

`<?xml standalone="no" ?>`

- Tài liệu XML không có liên kết với tài liệu DTD khác:

`<?xml version="1.0" standalone="yes" ?>`



# Thẻ chỉ thị xử lý

- **Thẻ chỉ thị xử lý**: cho phép mô tả thêm một số thông tin (liên quan xử lý) về tài liệu XML có ý nghĩa riêng với một công cụ xử lý nào đó.

- Dạng khai báo chung:

`<? Bo_xu_ly Du_lieu ?>`

**Bo\_xu\_ly** là ký hiệu của bộ xử lý sẽ tiến hành một số xử lý nào đó trên tài liệu XML . **Du\_lieu** là thông tin được gửi đến Bo\_xu\_ly.

# Thẻ chỉ thị xử lý

- Ví dụ:

```
<?xml-stylesheet type="text/css" href="Dinh_dang.css" ?>
```

- Là thẻ chỉ thị cần xử lý định dạng thể hiện tài liệu XML với “chương trình định dạng ” theo ngôn ngữ css được lưu trữ bên trong tập tin Dinh\_dang.css.

# Thẻ ghi chú

- **Thẻ ghi chú**: cho phép bổ sung các thông tin ghi chú có ý nghĩa đối với con người và hoàn toàn không có ý nghĩa với các hệ thống xử lý tài liệu XML.
- Cú pháp:  
**<-- Nội dung ghi chú -->**
- Chú ý:
  - Trong nội dung của ghi chú không có ký tự “-”.
  - Không nên đặt ghi chú trong 1 thẻ (*Thuộc giới hạn mở thẻ ... đóng thẻ*).
  - Không nên đặt ghi chú trước dòng khai báo **<?xml....?>**.

# Thẻ CDATA

- **CDATA (Unparsed Character Data)**: yêu cầu các bộ phân tích tài liệu XML Parser bỏ qua và không phân tích vào nội dung bên trong của thẻ này.
- Tác dụng của thẻ là cho phép sử dụng trực tiếp bên trong thẻ một số ký hiệu không được phép nếu sử dụng bên ngoài ( ví dụ các ký tự “<” , “>” , ...).
- Dạng khai báo chung:  
`<![CDATA [ Nội dung ]]>`

# PCDATA

- **PCDATA (Parsed Character Data):** là dữ liệu sẽ được đọc và phân tích bởi chương trình phân tích XML.
- Trong PCDATA không được phép dùng các ký tự đặc biệt có liên quan đến việc xác định các thành tố của XML như <, >, &, ...

# CDATA and PCDATA

```
<?xml version="1.0"?>
<!DOCTYPE employee SYSTEM "employee.dtd">
<employee>
<![CDATA[
  <firstname>vimal</firstname>
  <lastname>jaiswal</lastname>
  <email>vimal@javatpoint.com</email>
]]>
</employee>
```

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
▼ <employee>
  ▼ <![CDATA[
    <firstname>vimal</firstname> <lastname>jaiswal</lastname> <email>vimal@javatpoint.com</email>
  ]]>
</employee>
```

```
<?xml version="1.0"?>
<!DOCTYPE employee SYSTEM "employee.dtd">
<employee>
  <firstname>vimal</firstname>
  <lastname>jaiswal</lastname>
  <email>vimal@javatpoint.com</email>
</employee>
```

This XML file does not appear to have any style information associated with it.

```
▼ <employee>
  <firstname>vimal</firstname>
  <lastname>jaiswal</lastname>
  <email>vimal@javatpoint.com</email>
</employee>
```

# Cấu trúc tài liệu XML

- Khái niệm về **cấu trúc tài liệu XML**:
  - Tương ứng với cấu trúc của nội dung chính
  - Cách thức tổ chức, sắp xếp của các thẻ (có hay không có nội dung) trong nội dung chính.
- **Ngôn ngữ** đặc tả cấu trúc: Có rất nhiều ngôn ngữ đặc tả để mô tả cấu trúc tài liệu Xml như: DTD, XML Schema, XML- Data, Schematron , RELAX NG, v,v..  
.Trong số đó có 2 ngôn ngữ thông dụng là **DTD**, **XML Schema**.

# Cấu trúc tài liệu XML

- Đặc điểm của **DTD**:
  - Ra đời rất sớm
  - Cho phép mô tả văn bản **có cấu trúc bất kỳ**
  - Đơn giản, dễ học và sử dụng
  - Chỉ cho phép đặc tả một số “**kiểu dữ liệu đơn giản**” trong nội dung chính của tài liệu XML
- Đặc điểm của **XML Schema**:
  - Được đề xuất bởi W3C
  - Chỉ áp dụng cho **tài liệu XML**
  - Khó học và sử dụng so với DTD
  - Cho phép đặc tả **chi tiết về các “kiểu dữ liệu”** được sử dụng trong nội dung chính của tài liệu XML



# Cấu trúc tài liệu XML

- Ví dụ : Với tài liệu Xml:

```
<?xml version="1.0" encoding="utf-8"?>  
  <PHAN_SO>  
    <Tu_so> 4 </Tu_so>  
    <Mau_so> 3 </Mau_so>  
  </PHAN_SO>
```

# Cấu trúc tài liệu XML

- Đặc tả với **DTD**:

```
<!DOCTYPE PHAN_SO [  
  <!ELEMENT PHAN_SO (Tu_so, Mau_so) >  
  <!ELEMENT Tu_so #PCDATA >  
    <!-- Tu_so : Số nguyên // >0      -->  
  <!ELEMENT Mau_so #PCDATA>  
    <!-- Mau_so : Số nguyên // >0    -->  
>
```

# Sử dụng tên thẻ

- **Kĩ thuật sử dụng tên thẻ:** tên thẻ, tên các thuộc tính có 2 loại:
  - Tên không có tiền tố
  - Tên có tiền tố
- Tên **không tiền tố:** là **chuỗi** bao gồm các ký tự chữ ( a-z, A-Z), ký số (0-9) và một số ký tự khác như ‘—’ , “\_” , “ ” , . .

# Sử dụng tên thẻ

- Tên **có tiền tố**: có dạng 2 chuỗi ký tự cách nhau bởi ký tự ':' như sau:

**Chuoi\_tien\_to : Chuoi\_ten**

- Ví dụ :

**<A:MAT\_HANG ..../>**

**<B:MAT\_HANG .../>**

Thẻ **A:MAT\_HANG** tương ứng thông tin về mặt hàng trong công ty A.

Thẻ **B:MAT\_HANG** tương ứng thông tin về mặt hàng trong công ty B. 2 thẻ này có thể có các thuộc tính khác nhau.

# Sử dụng tên thẻ

- Sử dụng **tên có tiền tố** :
  - Nếu chỉ sử dụng tài liệu XML đơn lẻ, riêng cho ứng dụng cục bộ thì không cần thiết dùng tiền tố trong tên. Tuy nhiên nếu cần thiết **tiếp nhận, kết xuất toàn bộ/một phần tài liệu XML từ/đến một ứng dụng khác** ( rất thông dụng trong thương mại điện tử ) việc sử dụng tên với tiền tố là rất cần thiết.
  - Tiền tố của tên sẽ dùng **để phân biệt được nguồn gốc của một thẻ** trong tài liệu XML được tạo thành từ nhiều tài liệu XML khác **có các thẻ trùng phần tên không tiền tố**

# Sử dụng tên thẻ

- Một tài liệu XML có thành phần **<table>** dùng để mô tả đặc điểm của 1 **cái bàn** với các thuộc tính: length (*dài*), width (*rộng*), height (*cao*), material (*vật liệu*) ....
- Một tài liệu XML khác cũng có một thành phần tên là **<table>** nhưng dùng để mô tả một **bảng dữ liệu** với các thuộc tính: width (*bề rộng của bảng*), height (*chiều cao của bảng*) ....
- Khi hệ thống tiếp nhận cùng lúc cả 2 file XML này để lấy số liệu, rất khó để phân biệt các cấu trúc dữ liệu của XML.
- Do đó, người tạo XML phải mô tả **tên thành phần** và **thuộc tính** sao cho những thành phần này phải là **duy nhất trong mỗi cấu trúc XML** khi có sự tổng hợp thông tin từ nhiều nguồn khác nhau.

# Sử dụng tên thẻ

- XML mô tả thông tin của **cái bàn**:

```
<table length="2.5m" width="1.2m"  
        height="0.9m">  
  <name> Italian coffee style </name>  
  <material> training oval wood </material>  
</table>
```

- XML mô tả thông tin của **bảng dữ liệu**:

```
<table width="100%" height="80%">  
  <tr>  
    <td>Orange</td>  
    <td>Strawberry</td>  
  </tr>  
</table>
```

# Namespace

- **Namespace** giúp cho việc truy xuất đến các thành phần (*Element*) một cách tường minh.
- Namespace là tập hợp các tên dùng để cho phép kết hợp với các thành phần và thuộc tính bên trong một tài liệu XML nhằm giải quyết nguy cơ xung đột về tên của các phần tử khi thông tin được tổng hợp từ nhiều nguồn khác nhau.
- Thông qua Namespace, trình duyệt có thể kết hợp các file XML từ nhiều nguồn khác nhau, có thể truy xuất đến DTD để kiểm tra cấu trúc của XML nhận được có thực sự thích hợp, từ đó xác định được tính hợp lệ của XML tương ứng.



# Namespace

- Giải quyết xung đột:

```
<p:table length="2.5m" width="1.2m" height="0.9m">  
  <p:name> Italian coffee style </p:name>  
  <p:material> training oval wood  
  </p:material>  
</p:table>  
<s:table width="100%" height="80%">  
  <s:tr>  
    <s:td>Orange</s:td>  
    <s:td>Strawberry</s:td>  
  </s:tr>  
</s:table>
```

# Namespace

- Cú pháp khai báo namespace và thuộc tính **xmlns**:

```
<namespacePrefix:elementName      xmlns:namespacePrefix =  
"URI">
```

...

```
</namespacePrefix:elementName>
```

- **namespacePrefix**: phần viết tắt đại diện cho namespace được sử dụng như là tiền tố (prefix) cho các tag trong cùng nhóm.
- **xmlns**: là thuộc tính được sử dụng để khai báo và chỉ ra namespace cần thiết sẽ áp dụng trong cấu trúc XML.
- **URI** (*Uniform Resource Identifier*): chuỗi ký tự mô tả cho 1 nguồn tài nguyên nào đó duy nhất trên Internet.

# Namespace

- `<root>`  
`<p:table xmlns:p="http://www.w3.org/TR/html4/">`  
    `<p:tr>`  
        `<p:td>Apples</p:td>`  
        `<p:td>Bananas</p:td>`  
    `</p:tr>`  
    `</p:table>`  
  
`<s:table xmlns:s="https://www.w3schools.com/furniture">`  
    `<s:name>African Coffee Table</s:name>`  
    `<s:width>80</s:width>`  
    `<s:length>120</s:length>`  
    `</s:table>`  
  
`</root>`

# Namespace

- ```
<root xmlns:p="http://www.w3.org/TR/html4/"
xmlns:s="https://www.w3schools.com/furniture">
  <p:table>
    <p:tr>
      <p:td>Apples</p:td>
      <p:td>Bananas</p:td>
    </p:tr>
  </p:table>

  <s:table>
    <s:name>African Coffee Table</s:name>
    <s:width>80</s:width>
    <s:length>120</s:length>
  </s:table>
</root>
```

## 2. DTD

1. Đặc tả cấu trúc tài liệu XML với DTD
2. Đặc tả cấu trúc nội dung thẻ
3. Đặc tả thuộc tính của thẻ

# DTD

- Đặc tả cấu trúc tài liệu XML với DTD
- Có nhiều dạng khác nhau cho phép khai báo (đặc tả) cấu trúc của tài liệu XML:
- Dạng 1: Khai báo cấu trúc tài liệu XML được lưu trữ ngay bên trong chính tài liệu XML đó:

```
<!DOCTYPE Ten_the_goc [  
    đặc tả cấu trúc nội dung các thẻ  
    đặc tả thuộc tính các thẻ  

```

# DTD

- Dạng 2: Khai báo cấu trúc tài liệu XML được lưu trữ **bên ngoài dưới dạng một tập tin** chứa đặc tả cấu trúc nội dung các thẻ, đặc tả thuộc tính các thẻ. Cú pháp:

**<!DOCTYPE Ten\_the\_goc SYSTEM Ten\_tap\_tin >**

- Ví dụ :

**<!DOCTYPE DUONG\_TRON SYSTEM  
“DUONG\_TRON.dtd” >**

# DTD

- Dạng 3: Khai báo cấu trúc tài liệu XML đã được chuẩn hóa, có phạm vi sử dụng rộng rãi.

```
<!DOCTYPE Ten_the_goc PUBLIC Chuoi_nhan_dang  
>
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01  
Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
```



# Đặc tả cấu trúc nội dung thẻ

- Cú pháp chung đặc tả cấu trúc nội dung của một thẻ:

<!ELEMENT Ten\_the Bieu\_thuc\_dac\_ta  
cau\_truc\_noi\_dung >

- Bieu\_thuc có thể chỉ là một từ khoá
- Bieu\_thuc cũng có thể bao gồm nhiều từ khóa khác mô tả cách bố trí, sắp xếp các thành phần con bên trong thẻ
- Với A, B là 2 thẻ con của thẻ X:

A, B A, B sắp xếp theo thứ tự tuần tự A đến B

A\* A có thể lặp lại ít nhất 0 lần ( $\geq 0$ )

B+ B có thể lặp lại ít nhất 1 lần ( $\geq 1$ )

A? A có thể xuất hiện 0 hoặc 1 lần (0 or 1)

A|B Có thể chọn sử dụng A hay B

# Đặc tả cấu trúc nội dung thẻ DTD

- Từ khóa **ANY** : Thẻ có nội dung bất kì theo định chuẩn XML. Ví dụ :

**<!ELEMENT X ANY >**

- X có thể chứa nội dung bất kỳ, khai báo này chỉ để mô tả sự tồn tại của bên trong X một hoặc nhiều thẻ khác.

- Từ khóa **EMPTY** : Thẻ không có nội dung. Ví dụ :

**<!ELEMENT PHAN\_SO EMPTY >**

- PHAN\_SO không thể có nội dung mà chỉ có thể có các thuộc tính.

# Đặc tả cấu trúc nội dung thẻ DTD

- Từ khóa **#PCDATA** : Thẻ với nội dung là chuỗi văn bản.  
Ví dụ :

**<!ELEMENT Ho\_ten (#PCDATA) >**

- Ho\_ten có nội dung là chuỗi và không thể chứa các thẻ khác
- Với DTD muốn mô tả chi tiết hơn, dùng thẻ ghi chú. Ví dụ :

**<!ELEMENT He\_so (#PCDATA) >**

**<!-- He\_so : A\_Float -->**

# Đặc tả cấu trúc nội dung thẻ DTD

- **Dạng tuần tự:** Các thẻ con chỉ có thể xuất hiện 1 lần duy nhất và phải **theo đúng thứ tự** xuất hiện trong biểu thức

- Cú pháp :

**<!ELEMENT Ten\_the (Ten\_the\_1, Ten\_the\_2, ....) >**

- Ý nghĩa : The\_1, The\_2, ..., The\_k phải xuất hiện một lần duy nhất theo đúng thứ tự trên. Ví dụ:

**<!ELEMENT DON\_THUC(He\_so, So\_mu) >**

Thẻ DON\_THUC phải bao hàm bên trong 2 thẻ con He\_so, So\_mu theo đúng thứ tự trên

# Đặc tả cấu trúc nội dung thẻ DTD

- Ghi chú: Các thẻ bên trong có thể có tên trùng nhau. Ví dụ :

<!ELEMENT TAM\_GIAC (DIEM, DIEM, DIEM) >

- Có thể sử dụng từ khóa **#PCDATA** trong biểu thức tuần tự ( và các loại biểu thức khác ). Ví dụ :

<!ELEMENT X (#PCDATA, A, #PCDATA)>

- Thẻ X phải bao gồm 3 thành phần :
- Thành phần thứ 1 là chuỗi văn bản, thành phần thứ 2 là thẻ có tên A, thành phần thứ 3 là chuỗi văn bản

# Đặc tả cấu trúc nội dung thẻ DTD

- **Dạng tùy chọn:** Thẻ con có thể được sử dụng hay không sử dụng. Cú pháp ( dạng đơn giản) :

<!ELEMETNT Ten\_the ( Ten\_the\_con ?) >

- Thẻ đang xét có thể chứa 1 hay 0 lần xuất hiện của thẻ có tên là Ten\_the\_con

- Ví dụ :

<!ELEMETNT DON\_THUC (Ten?) >

Thẻ DON\_THUC có thể chứa hay không thẻ Ten

# Đặc tả cấu trúc nội dung thẻ DTD

- Có thể kết hợp với biểu thức tuần tự:

<!ELEMENT X (A,B?,C) >

- Có thể cho phép tùy chọn một tập hợp các thẻ:

<!ELEMENT X (A,B,C)? >

- X có thể chứa bên trong các thẻ A,B,C ( theo thứ tự trên ) hay cũng có thể không chứa bất kỳ thẻ nào
- **Dạng chọn** : Bắt buộc chọn một thẻ con để sử dụng trong tập hợp thẻ cho trước. Cú pháp:

<!ELEMENT:

Ten\_the(Ten\_the\_1|Ten\_the\_2|..|Ten\_the\_k) >

# Đặc tả cấu trúc nội dung thẻ DTD

- Có thể kết hợp với biểu thức tuần tự:

**<!ELEMENT X (A,B|C,D) >**

- Thành phần đầu tiên của thẻ X là thẻ A, kế đến là thẻ B hay thẻ C và thành phần cuối cùng phải là D

- Có thể cho phép chọn một tập hợp các thẻ

**<!ELEMENT X ( (A,B) | (C,D) ) >**

- X có thể bao hàm bên trong cặp thẻ A,B ( theo thứ tự trên ) hay cặp thẻ C,D ( theo thứ tự trên )



# Đặc tả cấu trúc nội dung thẻ DTD

- **Dạng lặp:**

- Dạng lặp ít nhất 0 lần : Các thẻ con có thể lặp lại nhiều lần hay có thể không có lần nào. Cú pháp :

`<!ELEMENT Ten_the (Ten_the_con*) >`

- Ý nghĩa: Thẻ đang xét có thể bao hàm bên trong nhiều thẻ có tên là Ten\_the\_con hay cũng có thể là thẻ rỗng ( không có nội dung )

- Ví dụ :

`<!ELEMENT LOP (HOC_SINH*) >`

- Thẻ LOP có thể chứa nhiều thẻ HOC\_SINH hay không có thẻ HOC\_SINH nào

# Đặc tả cấu trúc nội dung thẻ DTD

- Có thể mô tả lặp đồng thời nhiều thẻ con

**<!ELEMENT X (A,B,C)\* >**

- Các thẻ A,B,C theo thứ tự trên có thể lặp lại ít nhất 0 lần trong thẻ X

- Có thể kết hợp với biểu thức tuần tự. Ví dụ :

**<!ELEMENT X (A,B\*,C) >**

- Thẻ X có thành phần đầu tiên là thẻ A, kế đến có thể có nhiều hay 0 lần lặp của thẻ B và cuối cùng là thẻ C

# Đặc tả cấu trúc nội dung thẻ DTD

- Có thể kết hợp với biểu thức tùy chọn. Ví dụ :  
`<!ELEMENT X (A,B*,C?,D) >`

# Đặc tả cấu trúc nội dung thẻ DTD

- **Dạng lặp ít nhất 1 lần** : Các thẻ con có thể lặp lại nhiều lần và ít nhất là một lần. Cú pháp :

<!ELENEMT Ten\_the (Ten\_the\_con+) >

- Ý nghĩa : Thẻ đang xét có thể bao hàm bên trong ít nhất một thẻ có tên là Ten\_the\_con. Ví dụ :

<!ELEMENT DA\_THUC (DON\_THUC+) >

- Thẻ DATHUC phải bao hàm bên trong ít nhất một thẻ DON\_THUC

# Đặc tả cấu trúc nội dung thẻ DTD

- Có thể mô tả lặp đồng thời nhiều thẻ con

<!ELEMENT CT\_HOA\_DON  
(Mat\_hang,So\_luong,Don\_gia) + >

- Các thẻ CT\_HOA\_DON phải bao hàm ít nhất 1 lần 3 thẻ Mat\_hang,So\_luong,Don\_gia

- Có thể kết hợp với biểu thức tuần tự. Ví dụ

<!ELEMENT DA\_GIAC  
(DIEM,DIEM,DIEM+) >

- Các thẻ DA\_GIAC phải bao hàm ít nhất 3 thẻ DIEM

# Đặc tả thuộc tính của thẻ DTD

- Cú pháp khai báo **đặc tả thuộc tính** chung:

<!ATTLIST Ten\_the

Ten\_thuoc\_tinh\_1 Kieu\_1 Tham\_so\_1

Ten\_thuoc\_tinh\_2 Kieu\_2 Tham\_so\_2

...

Ten\_thuoc\_tinh\_k Kieu\_k Tham\_so\_k

>

# Đặc tả thuộc tính của thẻ DTD

- Ý nghĩa :
- **Ten\_the** : tên thẻ cần khai báo các thuộc tính
- **Ten\_thuoc\_tinh\_1, Ten\_thuoc\_tinh\_2, ... Ten\_thuoc\_tinh\_k** : Tên các thuộc tính của thẻ
- **Kieu\_1, Kieu\_2, ..., Kieu\_k** : Mô tả tập hợp các giá trị mà thuộc tính có thể nhận
- **Tham\_so\_1, Tham\_so\_2, ..., Tham\_so\_k** : Mô tả một số tính chất trên thuộc tính tương ứng

# Đặc tả thuộc tính của thẻ DTD

- Ví dụ: Đặc tả cấu trúc tài liệu XML biểu diễn thông tin về biểu thức phân số:

$$P = 4/5 + 6/7 * 1/3 - 10/3 + 11/2 * 2/3$$



# Đặc tả thuộc tính của thẻ DTD

```
<?xml version="1.0" encoding="utf-8"?>
  <!DOCTYPE BIEU_THUC [
    <!ELEMENT BIEU_THUC (PHAN_SO | TICH_SO)+ >
      <ATTLIST BIEU_THUC Ten CDATA #IMPLIED
        <!-- Ten : A_String -->
      >
    <!ELEMENT PHAN_SO EMPTY >
      <ATTLIST PHAN_SO
        Tu_so CDATA #REQUIRED
        <!-- Tu_so : A_Int -->
        Mau_so CDATA #REQUIRED
        <!-- Mau_so : A_Int // >0 -->
      >
    <!ELEMENT TICH_SO (PHAN_SO)+ >
  ]>
```

# Đặc tả thuộc tính của thẻ DTD

- **Kiểu thuộc tính** : Mô tả **tập hợp các giá trị** của thuộc tính. Một số cách thông dụng mô tả:

- Cách 1 : Dùng từ khoá **CDATA**

Cú pháp :

**<!ATTLIST Ten\_the**

...

**Ten\_thuoc\_tinh      CDATA**

...

**>**

- Tập hợp các giá trị của thuộc tính với khai báo CDATA chính là tập hợp các **chuỗi**

# Đặc tả thuộc tính của thẻ DTD

- Ví dụ : Đặc tả cấu trúc tài liệu XML biểu diễn phương trình đường thẳng trong mặt phẳng

<?xml version="1.0" encoding="utf-8"?>

<!DOCTYPE DUONG\_THANG [

<!ELEMENT DUONG\_THANG EMPTY>

<!--phương trình  $ax + by + c = 0$  -->

# Đặc tả thuộc tính của thẻ DTD

```
<ATTLIST DUONG_THANG
  Ten  CDATA      #IMPLIED
      <!-- Ten : A_String  -->
  a    CDATA      #REQUIRED
      <!-- a : A_Float     -->
  b    CDATA      #REQUIRED
      <!-- b : A_Float     -->
  c    CDATA      #REQUIRED
      <!-- c : A_Float     -->
>
      <!--a,b không đồng thời là 0      -->
]>
```

# Đặc tả thuộc tính của thẻ DTD

- Cách 2: dùng **biểu thức liệt kê**. Cú pháp :  
**<!ATTLIST** Ten\_the  
...  
Ten\_thuoc\_tinh ( Gia\_tri\_1,Gia\_tri\_2,....\_gia\_tri\_k) ...  
**>**
- Ý nghĩa : Tập hợp các giá trị có thể có của thuộc tính đang xét chính là tập hợp các giá trị được liệt kê
- **Gia\_tri\_1,Gia\_tri\_2, ....,Gia\_tri\_k**: Các giá trị này là các chuỗi ký tự

# Đặc tả thuộc tính của thẻ DTD

- Ví dụ: Đặc tả cấu trúc tài liệu XML biểu diễn thông tin về phiếu điểm của một học sinh:

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<!DOCTYPE PHIEU_DIEM [
```

```
<!ELEMENT PHIEU_DIEM (HOC_SINH, DIEM_SO+ )  
>
```

# Đặc tả thuộc tính của thẻ DTD

<!ELEMENT HOC\_SINH EMPTY >

<ATTLIST HOC\_SINH

Ho\_ten CDATA #REQUIRED

<!-- Ho\_ten : A\_String -->

Ngay\_sinh CDATA #REQUIRED

<!--Ngay\_sinh : A\_Date -->

Xep\_loai(“Giỏi”, “Khá”, “Trung bình”, “Yếu”)#IMPLIED

>

<!ELEMENT DIEM\_SO EMPTY >

<ATTLIST DIEM

Ten\_mon CDATA #REQUIRED

<!-- Ten\_mon : A\_String -->

Gia\_tri CDATA #REQUIRED

<!-- Gia\_tri : A\_Float // từ 0 đến 10 -->

## Đặc tả thuộc tính của thẻ

- **Tham số của thuộc tính**: mô tả tính chất của thuộc tính, một số cách mô tả:
- Cách 1: Dùng từ khóa **#REQUIRED**. Cú pháp :  
<!ATTLIST Ten\_the

...

Ten\_thuoc\_tinh Kieu **#REQUIRED**

...

>

- Ý nghĩa : Thuộc tính đang xét là thuộc tính **bắt buộc** phải có. Đây là cách sử dụng phổ biến nhất



# Đặc tả thuộc tính của thẻ DTD

- Ví dụ : Đặc tả cấu trúc tài liệu XML biểu diễn thông tin về các đơn thức với tên bắt buộc phải có

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE DON_THUC [
  <!ELEMENT DON_THUC (He_so, So_mu ) >
  <ATTLIST DON_THUC
    Ten      CDATA #REQUIRED
             <!-- Ten : A_String      -->
    Bien_so  CDATA #REQUIRED
             <!-- Bien_so: A_String   -->
  >
  <!ELEMENT He_so #PCDATA >
             <!-- He_so : A_Float     -->
  <!ELEMENT So_mu #PCDATA >
             <!-- So_mu : A_Int // >=0 -->
]>
```

# Đặc tả thuộc tính của thẻ DTD

- Cách 2 : Dùng từ khóa **#IMPLIED**. Cú pháp :  
<!ATTLIST Ten\_the

...

Ten\_thuoc\_tinh Kieu **#IMPLIED**

...

>

- Ý nghĩa : Thuộc tính đang xét là **tùy chọn và không bắt buộc phải có**.

# Đặc tả thuộc tính của thẻ DTD

- Ví dụ : Đặc tả cấu trúc tài liệu XML biểu diễn thông tin về tam thức  $P(x) = 2x^2 - 4x + 6$ .

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<!DOCTYPE TAM_THUC [
```

```
<!ELEMENT TAM_THUC
```

```
(DON_THUC,DON_THUC,DON_THUC) >
```

```
<ATTLIST TAM_THUC
```

```
    Ten      CDATA#IMPLIED
```

```
    <!-- Ten : A_String
```

```
    <!-- Bien_so: A_String // sinh sẵn là x -->
```

```
>
```

# Đặc tả thuộc tính của thẻ DTD

<!ELEMENT DON\_THUC EMPTY >

<ATTLIST DON\_THUC

He\_soCDATA #REQUIRED

<!-- He\_so : A\_Float // Khác 0 nếu

So\_mu=2 -->

So\_mu (0,1,2)#REQUIRED

<!-- So\_mu : A\_Int // =0,1,2 và khác nhau

-->

>

# Đặc tả thuộc tính của thẻ DTD

Cách 3: Dùng từ khóa **#FIXED**. Cú pháp :

```
<!ATTLIST Ten_the
```

```
...
```

```
Ten_thuoc_tinh Kieu      #FIXED Gia_tri
```

```
...
```

```
>
```

- Ý nghĩa : Thuộc tính đang xét phải có giá trị cố định là **Gia\_tri**. Trường hợp này ít được sử dụng

# Đặc tả thuộc tính của thẻ DTD

- Ví dụ: Đặc tả cấu trúc tài liệu XML biểu diễn thông tin về các đơn thức chỉ với biến số x:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE DON_THUC [
  <!ELEMENT DON_THUC (He_so, So_mu ) >
  <ATTLIST DON_THUC
    Ten      CDATA #REQUIRED
             <!-- Ten : A_String      -->
    Bien_so  CDATA  #FIXED "x"
             <!-- Bien_so: A_String    -->
  ]
>
```

# Đặc tả thuộc tính của thẻ DTD

```
<!ELEMENT He_so (#PCDATA) >
```

```
  <!-- He_so : A_Float-->
```

```
<!ELEMENT So_mu (#PCDATA) >
```

```
  <!-- So_mu : A_Int // >=0  -->
```

# XML Schema

- **XML Schema**: khai báo XML Schema là tạo lập tài liệu XML mà nội dung chính là **các thẻ đánh dấu**, các thẻ này **mô tả cho cấu trúc các thẻ** của một tài liệu XML khác. Cú pháp:

```
<?xmlversion="1.0"encoding="utf-8"?>
```

```
<xs:schema
```

```
xmlns:xs="http://www.w3.org/2001/XMLSchema">
```

```
    Đặc tả các thẻ
```

```
    Đặc tả các kiểu
```

```
</xs:schema>
```



# XML Schema

- Thông tin về một thẻ được mô tả tập trung qua một phương cách duy nhất là **kiểu**.
- Mỗi thẻ sẽ có tương ứng một kiểu, đặc tả kiểu mô tả kiểu của thẻ cùng với một số thông tin khác chính là **cách sắp xếp các thành phần bên trong** của thẻ và hệ thống **các thuộc tính** của thẻ.

# XML Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="DA_THUC" type="K_DA_THUC"/>

  <xs:complexType name="K_DA_THUC">
    <xs:sequence>
      <xs:element name="DON_THUC"
type="K_DON_THUC" minOccurs="1"/>
    </xs:sequence>
    <xs:attribute name="Ten" type="xs:string" />
    <xs:attribute name="Bien_so" type="xs:string"/>
  </xs:complexType>
```

# XML Schema

- Tài liệu XML có thể gốc là **DA\_THUC** thẻ này có kiểu là **kiểu phức hợp** tên là **K\_DA\_THUC** ( có thể dùng cùng tên là DA\_THUC).
- Kiểu phức hợp K\_DA\_THUC bao gồm bên trong:
  - Thẻ **DON\_THUC** có kiểu là kiểu phức hợp và thẻ DON\_THUC phải **xuất hiện ít nhất 1 lần**
  - 2 thuộc tính :
    - **Ten** với kiểu là kiểu cơ sở dạng **chuỗi**
    - **Bien\_so** với kiểu là kiểu cơ sở dạng **chuỗi**

# XML Schema

```
<xs:complexType name="K_DON_THUC">  
  <xs:attribute name="He_so" type="xs:float"/>  
  <xs:attribute name="So_mu"  
    type="SO_TU_NHIEN"/>  
</xs:complexType>  
<xs:simpleType name="SO_TU_NHIEN">  
  <xs:restriction base="xs:int">  
    <xs:minInclusive value="0"/>  
  </xs:restriction>  
</xs:simpleType>  
</xs:schema>
```

# XML Schema

- Kiểu phức hợp K\_DON\_THUC không có nội dung và chỉ bao gồm các thuộc tính:
  - He\_so có kiểu là kiểu cơ sở loại số thực
  - So\_mu có kiểu là kiểu đơn giản với tên SO\_TU\_NHIEN
- Kiểu đơn giản SO\_TU\_NHIEN chính là kiểu cơ sở số nguyên với hạn chế: giá trị phải lớn hơn hay bằng 0

# XML Schema Đặc tả kiểu

- XML Schema có 3 loại kiểu chính :
  - Loại 1 : Kiểu định nghĩa sẵn (BuiltInType)
  - Loại 2 : Kiểu đơn giản (simpleType)
  - Loại 3 : Kiểu phức hợp (complexType).

# XML Schema kiểu định nghĩa sẵn

Ten_kieu_co_so	Ý nghĩa
string	Chuỗi ký tự
int, integer	Số nguyên
float	Số thực chính xác đơn
double	Số thực chính xác kép
boolean	Giá trị logic
date	ngày
month	Tháng
ID	Chuỗi định danh
binary	Dữ liệu nhị phân

# XML Schema kiểu định nghĩa sẵn

- Kiểu định nghĩa sẵn – thư viện
- Ý nghĩa sử dụng :
  - Được sử dụng để mô tả trực tiếp kiểu của các thuộc tính hay của thẻ thỏa mãn 2 điều kiện :
    - Điều kiện 1 : thẻ không có thuộc tính
    - Điều kiện 2 : thẻ không chứa thẻ khác (nội dung là chuỗi văn bản) và có miền giá trị (tập hợp giá trị có thể có) thích hợp với kiểu



# XML Schema kiểu định nghĩa sẵn

- Cú pháp :

- Khi dùng với thẻ:

- ```
<xs:element name="Ten_the" type="Ten_kieu_co_so" ... />
```

- Khi dùng với thuộc tính:

- ```
<xs:attribute name="Ten_thuoc_tinh" type="Ten_kieu_co_so" .. />
```

- Ví dụ:

```
<xs:element name="Ho_ten" type="xs:string" />
```

```
<xs:element name="Ngày_sinh" type="xs:date" />
```

```
<xs:attribute name="He_so" type="xs:float"/>
```

```
<xs:attribute name="f" type="xs:boolean"/>
```

# XML Schema kiểu đơn giản

- Kiểu đơn giản (**simpleType**): Là các kiểu do người dùng định nghĩa dựa trên các kiểu cơ sở có sẵn.
- Ý nghĩa sử dụng: để mô tả trực tiếp kiểu của các thuộc tính hay các thẻ thỏa mãn:
  - Điều kiện 1 : Không có thuộc tính
  - Điều kiện 2 : Không chứa thẻ khác ( nội dung là chuỗi văn bản) và có miền giá trị ( tập hợp giá trị có thể có ) là tập con của miền giá trị một kiểu cơ sở nào đó

# XML Schema kiểu đơn giản

- Cú pháp:

`<xs:simpleType name="Ten_kieu">`

`<xs:restriction base="Ten_kieu_co_so">`

Giới hạn ( ràng buộc ) trên miền giá trị

`</xs:restriction>`

`</xs:simpleType>`

# XML Schema kiểu đơn giản

- Khai báo cận dưới: Sử dụng từ khoá **minInclusive** ( cận dưới cho phép sử dụng biên ), **minExclusive** ( cận dưới không cho phép sử dụng biên)
- Khai báo cận trên: Sử dụng từ khoá **maxInclusive** ( cận trên cho phép sử dụng biên), **maxExclusive** ( cận trên không cho phép sử dụng biên)

# XML Schema kiểu đơn giản

```
<xs:simpleType name="KY_SO">  
  <xs:restriction base="xs:int">  
    <xs:minInclusive value="0" />  
    <xs:maxInclusive value="9" />  
  </xs:restriction>  
</xs:simpleType>
```

```
<xs:simpleType name="DIEM_SO">  
  <xs:restriction base="xs:float">  
    <xs:minInclusive value="0" />  
    <xs:maxInclusive value="10" />  
  </xs:restriction>  
</xs:simpleType>
```

# XML Schema kiểu đơn giản

- Giới hạn loại **liệt kê** trên kiểu cơ sở: Cho phép xác định miền giá trị của kiểu đơn giản bằng cách liệt kê các giá trị. Cú pháp:

```
<xs:simpleType name="Ten_kieu">  
  <xs:restriction  
    base="Ten_kieu_co_so_loai_so">  
    <xs:enumeration value="Gia_tri_1" />  
    <xs:enumeration value="Gia_tri_2" />  
    ...  
    <xs:enumeration value="Gia_tri_k" />  
  </xs:restriction>  
</xs:simpleType>
```

# XML Schema kiểu đơn giản

- Ví dụ:

```
<xs:simpleType name="LOAI_HOC_LUC" >  
  <xs:restriction base="xs:string">  
    <xs:enumeration value="Giỏi" />  
    <xs:enumeration value="Khá" />  
    <xs:enumeration value="Trung bình" />  
    <xs:enumeration value="Yếu" />  
  </xs:restriction>  
</xs:simpleType>
```

# XML Schema kiểu phức hợp

- Kiểu phức hợp **complexType**: Là các kiểu do người dùng tự định nghĩa cho phép mô tả nội dung và các thuộc tính của các thẻ được khai báo thuộc về kiểu đang xét.
- Ý nghĩa sử dụng :
  - Được sử dụng để mô tả kiểu của các thẻ thỏa một trong 2 điều kiện :
    - Điều kiện 1 : Có thuộc tính
    - Điều kiện 2 : Có chứa thẻ khác



# XML Schema kiểu phức hợp

- Cú pháp chung:

```
<xs:complexType name="Ten_kieu">
```

```
Dac_ta_cau_truc_noi_dung      Dac_ta_thuoc_tinh
```

```
</xs:complexType>
```

# XML Schema kiểu phức hợp

- **Dac\_ta\_cau\_truc\_noi\_dung** : Mô tả cách thức tổ chức, sắp xếp các thẻ con bên trong. Tương tự như DTD, XML Schema cho phép nhiều dạng tổ chức sắp xếp (tuần tự, chọn, lặp) các thẻ con. Ngoài ra cho phép khai báo chi tiết hơn về số lần lặp của một thành phần.
- **Dac\_ta\_thuoc\_tinh**: Mô tả hệ thống các thuộc tính của thẻ. Việc mô tả các thuộc tính trong XML Schema cũng tương tự như mô tả thuộc tính trong DTD nhưng cho phép định nghĩa và sử dụng các kiểu đơn giản để mô tả chi tiết về miền giá trị của một thuộc tính.

# XML Schema kiểu phức hợp

- Đặc tả cấu trúc nội dung dạng **tuần tự**:
- Dạng tuần tự : Sử dụng thẻ/từ khóa **sequence**. Cú pháp :

```
<xs:complexType name="Ten_kieu">
```

```
  <xs:sequence> Thanh_phan_1
```

```
                Thanh_phan_2
```

```
                Thanh_phan_k
```

```
  </xs:sequence>
```

```
  ....
```

```
</xs:complexType>
```

# XML Schema kiểu phức hợp

- Ví dụ:

```
<xs:complexType name="DIEM">  
  <xs:sequence>  
    <xs:element name="x" type="xs:float" />  
    <xs:element name="y" type="xs:float" />  
  </xs:sequence>  
</xs:complexType>
```

# XML Schema kiểu phức hợp

- Đặc tả cấu trúc nội dung dạng **tùy chọn**:
- Dạng tùy chọn : Sử dụng thẻ/từ khóa **choice**

Cú pháp :

```
<xs:complexType name="Ten_kieu">
```

```
  <xs:choice> Thanh_phan_1  
                Thanh_phan_2  
                Thanh_phan_k
```

```
  </xs:choice>
```

....

```
</xs:complexType>
```

# XML Schema kiểu phức hợp

- Ví dụ :

```
<xs:complexType name="X">
```

```
<xs:choice>
```

```
  <xs:element name="A" type="A" />
```

```
  <xs:element name="B" type="xs:string" />
```

```
</xs:choice>
```

```
</xs:complexType>
```

# XML Schema kiểu phức hợp

- Đặc tả cấu trúc nội dung dạng **lặp**:
- Dạng lặp : Sử dụng thuộc tính/từ khóa **minOccurs** , **maxOccurs**.  
Cú pháp:

```
<xs:complexType name="Ten_kieu">
  <xs:sequence>
    ...
    <xs:element name="Ten_the_con" type="Kieu_the_con"
      minOccurs="So_lan_lap_toi_thieu"
      maxOccurs="So_lan_lap_toi_da"      />
    ...
  </xs:sequence>
  ....
</xs:complexType>
```

# XML Schema kiểu phức hợp

- Ví dụ:

```
<xs:complexType name="DA_THUC">  
  <xs:sequence>  
    <xs:element name="DON_THUC"  
      type="DON_THUC" minOccurs="1" />  
  </xs:sequence>  
  <-- Mô tả các thuộc tính -->  
  ...  
</xs:complexType>
```



# XML Schema kiểu phức hợp

```
<xs:complexType name="DA_GIAC">  
  <xs:sequence>  
    <xs:element name="DIEM" type="DIEM"  
      minOccurs="3"  
      maxOccurs="3" />  
  </xs:sequence>  
  <-- Mô tả các thuộc tính -->  
  ...  
</xs:complexType>
```

# XML Schema kiểu phức hợp

- Đặc tả **thuộc tính**: cho phép mô tả hệ thống các thuộc tính của một thẻ
- Cú pháp :

```
<xs:complexType name="Ten_kieu">
```

Đặc tả cấu trúc nội dung

....

```
<xs:attribute name="Ten_thuoc_tinh"
```

```
type="Kieu_thuoc_tinh" Tinh_chat_thuoc_tinh />
```

....

```
</xs:complexType>
```

# XML Schema kiểu phức hợp

- **Tinh\_chat\_thuoc\_tinh** : Mô tả một số tính chất của thuộc tính, mỗi tính chất tương ứng với một từ khóa riêng.

```
<xs:attribute name="Ten_thuoc_tinh"  
    type="Kieu_thuoc_tinh"  
    Tu_khoa_1="Gia_tri_1"  
    Tu_khoa_2="Gia_tri_2"  
    Tu_khoa_k="Gia_tri_k" />
```

- Một số tính chất thông dụng:
  - Giá trị định sẵn : từ khóa **default**
  - Giá trị cố định: từ khóa **fixed**
  - Tùy chọn (có hay không có) sử dụng : từ khóa **use**

# XML Schema kiểu phức hợp

- **Đặc tả thẻ**: các thông tin cần mô tả khi đặc tả một thẻ trong XML bao gồm:
  - Tên thẻ
  - Kiểu của thẻ
  - Một số tính chất khác của thẻ
- Cú pháp khai báo:  

```
<xs:element name="Ten_the" type="Ten_kieu"  
Thuoc_tinh_khac />
```

# XML Schema kiểu phức hợp

- Tên của kiểu: mô tả thông tin về thẻ, **tên kiểu và tên thẻ được đặt trùng nhau.**
- Thuộc tính của thẻ: mô tả các tính chất của thẻ, thông dụng nhất là **minOccurs**, **maxOccurs** .
- Khi đặc tả các thẻ vấn đề quan trọng nhất là **xác định loại kiểu** sẽ dùng trong thẻ.

# XML Schema kiểu phức hợp

- **Thuật giải đặc tả thẻ**: Xét loại kiểu của A:  
A là **kiểu phức hợp**, đặc tả kiểu phức hợp A bao gồm:
  - Đặc tả **hệ thống các thẻ con** của thẻ gốc X
    - Đặc tả thẻ X1 với thông tin về kiểu (giả sử là A1)
    - Đặc tả thẻ X2 với thông tin về kiểu (giả sử là A2)
    - ...
    - Đặc tả thẻ XK với thông tin về kiểu (giả sử là Ak)
  - Đặc tả **hệ thống các thuộc tính** của thẻ gốc X
    - Đặc tả thuộc tính T1 với thông tin về kiểu (giả sử là B1)
    - Đặc tả thuộc tính T2 với thông tin về kiểu (giả sử là B2) ...
    - Đặc tả thuộc tính Tk với thông tin về kiểu (giả sử là Bk)

# XML Schema kiểu phức hợp

- A là **kiểu đơn giản**: Đặc tả kiểu đơn giản A bao gồm
  - Đặc tả kiểu cơ sở của A
  - Đặc tả các hạn chế trên kiểu cơ sở của A

# XML Schema kiểu phức hợp

- A là **kiểu cơ sở** :

Không cần Đặc tả thêm

Xét loại kiểu của A1

Xét loại kiểu của A2

...

Xét loại kiểu của Ak

Xét loại kiểu của B1

Xét loại kiểu của B2

...

Xét loại kiểu của Bk

Xét loại kiểu của T1

Xét loại kiểu của T2

...

Xét loại kiểu của Tk

.....

Xét loại kiểu của các kiểu phát sinh thêm khi đặc tả các kiểu phía trên

.....



# Tài liệu tham khảo

- [www.w3schools.com/xml](http://www.w3schools.com/xml)
- <http://www.ibm.com> – XML là gì
- Sách: Công nghệ XML và ứng dụng – Tác giả: Nguyễn Tiến Huy
- Charles F. Goldfarb and Paul Prescod. XML Handbook™, Fifth Edition. Prentice Hall, December, 2003



# Bài tập trên lớp 1 – Chuyển đổi

```
<?xml version="1.0" encoding="UTF-8"?>
<Danh_muc_sach>
  <Sach id="S0001">
    <Tac_gia>Nguyen Kim Anh</Tac_gia>
    <Ten_sach>Nguyen ly cac he co so du lieu</Ten_sach>
    <Nganh>Cong Nghe Thong Tin</Nganh>
    <Gia>32.000</Gia>
    <NXB>Dai Hoc Quoc Gia Ha Noi</NXB>
    <Xuat_ban>2004</Xuat_ban>
    <So_luong_ban>120</So_luong_ban>
  </Sach>
  <Sach id="S0002">
    .....
  </Sach>
  .....
</Danh_muc_sach>
```

# Bài tập trên lớp 2 – Sửa các lỗi

```
<!-- This is about movies! -->
<!ELEMENT actor ( #PCDATA )>
<!ELEMENT movie (actor*, title, year*)>
<!ELEMENT title ( #PCDATA )>
<!ELEMENT year ( #PCDATA )>

<movie id = 1>
  <title>The Quest</title>
  <actor>Tom Smith</actor>
  <year>2007</year>
<movie id = 2>
  <title>Summer</title>
  <year>1998</year>
  <actor>Susie Black</actor>
  <actor>Paul White</actor>
</movie>
<movie id = 3><title>Hello World</movie></title>
```

# Bài tập 3 – Xác định DTD, XML Schema

```
<PhieuDiem>
  <SinhVien>
    <HoTen>Nguyen Van
Quan</HoTen>
    <Lop>KTPMK10B</Lop>
    <DiaChi>Bac Ninh</DiaChi>
    <GioiTinh>Nam</GioiTinh>
    <NgaySinh>31/12/1992</NgaySin
h>
  </SinhVien>

  <BangDiem>
    <MonHoc>
      <STT>1</STT>
      <TenMon>XML</TenMon>
      <DienKy1>10</DienKy1>
      <DienKy2>10</DienKy2>
      <DienTB>10</DienTB>
    </MonHoc>

    <MonHoc>
      <STT>2</STT>
      <TenMon>Java</TenMon>
      <DienKy1>10</DienKy1>
      <DienKy2>10</DienKy2>
      <DienTB>10</DienTB>
    </MonHoc>
  </BangDiem>
</PhieuDiem>
```

## Bài tập 4

- Một tài liệu XML được dùng để biểu diễn kết quả học tập của nhiều sinh viên. Mỗi sinh viên được mô tả bởi:
- Thông tin cá nhân sinh viên: bao gồm các thông tin về Mã số sinh viên, Họ tên, Ngày sinh, Lớp, Trạng thái học tập
- Bảng điểm sinh viên: bao gồm các thông tin về kết quả học tập của từng học phần. Mỗi học phần có thông tin về Học kỳ, Mã học phần, Tên học phần, Tín chỉ, Mã lớp học, Điểm Quá trình, Điểm Thi, Điểm chữ
- Hãy viết tài liệu DTD (Document Type Definition) để đặc tả tài liệu XML trên.