

Công nghệ XML và WEB ngữ nghĩa

The Semantic WEB

Giáo viên

- Trần Nguyên Ngọc
- Nơi công tác: bộ môn Khoa học máy tính, Khoa CNTT, HVKTQS
- Hướng nghiên cứu: Xử lý tín hiệu hình ảnh, Trí tuệ nhân tạo, Lý thuyết điều khiển tối ưu, Khai phá dữ liệu
- Liên hệ: email tnn1999@mail.ru cell. 0948435163; tầng 2 nhà A1
- Lịch gặp sinh viên: Thứ 4 hàng tuần 14h15 đến 14h45 tại nhà A1

Môn học

- Công nghệ XML và Web ngữ nghĩa- ***XML Technologies and the Semantic Web*** thuộc nhóm chuyên ngành Khoa học máy tính
- Số buổi học: 15
- Thi: Vấn đáp trên bài tập lớn
- Các môn học liên quan: CTDL>, Multimedia, Trí tuệ nhân tạo, Lập trình mạng, Khai phá dữ liệu

Các nội dung chính của môn học

- Lịch sử hình thành và nhu cầu phát triển web ngữ nghĩa.
- Ontology nền tảng xây dựng Web ngữ nghĩa
- Giới thiệu Ngôn ngữ truy vấn SPARQL
- Công nghệ XML và các xu hướng phát triển
- Bài toán Xây dựng hệ thống web ngữ nghĩa
- Webcrawler và khai thác thông tin từ Web ngữ nghĩa

Tài liệu cơ bản

- Google “ Semantic Web”
- Michael C. Daconta, Leo J. Obrst, Kevin T. Smith. *The Semantic Web - A Guide to the Future of XML, Web Services, and Knowledge Management*, Wiley – 2003.
- T. Berners-Lee, J. Hendler, O. Lassila, *The Semantic Web*, *Scientific American*, May 2001.
- D. Brickley, R.V. Guha, *Resource Description Framework (RDF) Schema Specification*, World Wide Web Consortium, Proposed recommendation 2001.

Tài liệu đọc thêm

- <http://www.w3.org/TR/rdf-sparql-query/>
- SPARQL Query Language for RDF -
[http://www.w3schools.com/rdf/default.asp.](http://www.w3schools.com/rdf/default.asp)
- <http://razor.occams.info/semweb>. (Thư viện semweb)
- RDF – Resource Description Framework
- <http://www.w3.org/TR/rdf-primer/>
- OWL Web Ontology Language
- <http://www.w3.org/TR/owl-features/>
- <http://protege.stanford.edu/>

Một số yêu cầu

- Chuẩn bị tài liệu tham khảo
- Trang bị kỹ năng tìm tài liệu với Google
- Cài đặt thư viện và chuẩn bị môi trường lập trình
- Có điểm chuyên cần >5, điểm thường xuyên>4
- Tích cực trao đổi, làm việc theo nhóm

Giới thiệu XML

Extensible Markup Language

Mục tiêu

- Giới thiệu tổng quan về ngôn ngữ XML
- Làm quen với một số môi trường soạn thảo XML đơn giản

Giới thiệu

- XML(Extensible Markup Language): ngôn ngữ định dạng mở rộng.
- XML là ngôn ngữ được định nghĩa bởi tổ chức mạng toàn cầu(World Wide Web Consortium) thường được viết tắt W3C.
- XML là một ngôn ngữ tổng quát dùng để định nghĩa dữ liệu thông qua các thẻ.
- XML là một chuẩn không phụ thuộc vào bất kì một hệ điều hành nào.

Ngôn ngữ định dạng (markup Language)

- Ngôn ngữ định dạng là tất cả những gì dùng để mô tả nội dung một tài liệu.

- Ví dụ

```
<html>
```

```
<head>
```

```
<title>Chào các bạn đến với thế giới của HTML</title>
```

```
</head>
```

```
<body>
```

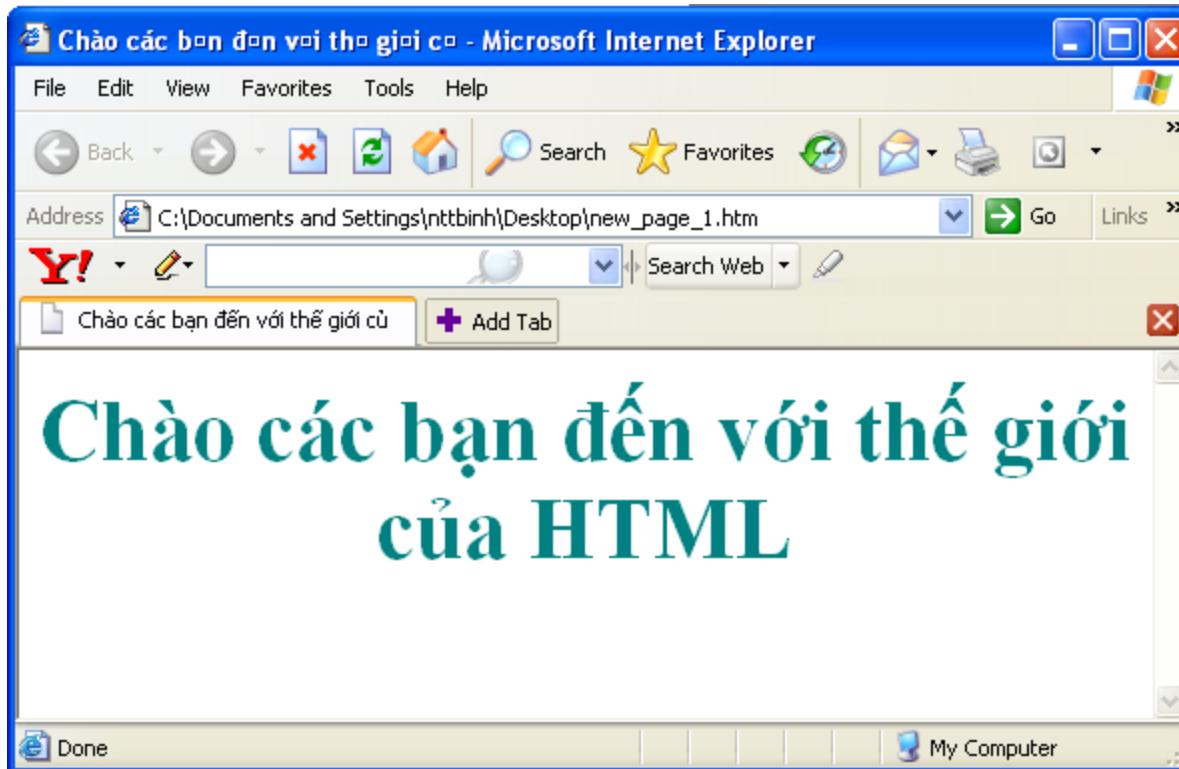
```
<h1 align="center"><font color="#008080">Chào các bạn đến  
với thế giới của HTML</font></h1>
```

```
</body>
```

```
</html>
```

Ngôn ngữ định dạng (Markup Language)

- Kết quả của trang HTML trên trình duyệt IE.



Ngôn ngữ định dạng (Markup Language)

- HTML thực hiện định dạng bằng các thẻ(tags) như <head>, <Center>...
- Thẻ chỉ cho trình duyệt biết cách hiển thị nội dung tài liệu.
- Tất cả những gì mà ngôn ngữ định dạng thể hiện là cung cấp thông tin và cách thức trình bày nội dung tài liệu.

HTML và XML

- Cả hai đều là ngôn ngữ định dạng (định dạng theo nghĩa cách quy định để xử lý và chứa nội dung tài liệu).
- HTML sử dụng các thẻ được định nghĩa và quy định sẵn.
- XML đưa ra một số quy tắc cho phép người dùng tự định nghĩa các thẻ.

Ngôn ngữ XML

- Ví dụ:

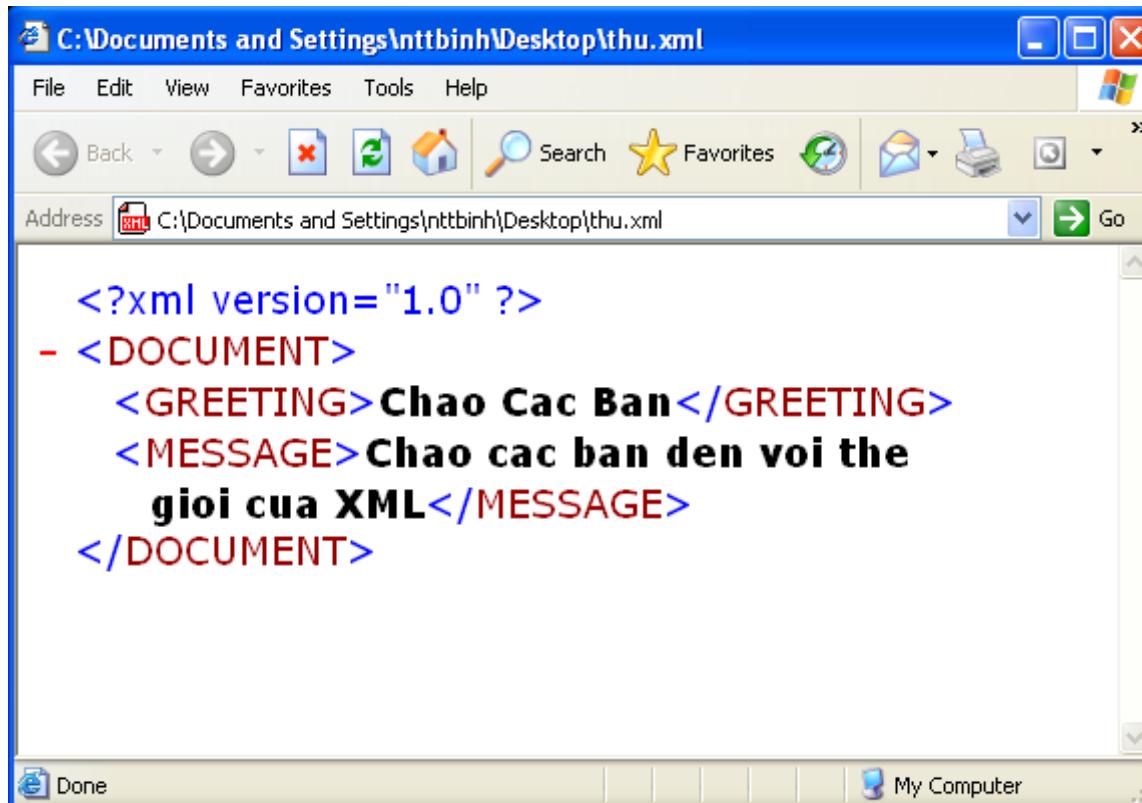
```
<?xml version="1.0"?>
<DOCUMENT>
    <GREETING>
        Chao Cac Ban
    </GREETING>
    <MESSAGE>
        Chao cac ban den voi the gioi cua XML
    </MESSAGE>
</DOCUMENT>
```

Ngôn ngữ XML

- Tất cả các chỉ thị của XML đều được bắt đầu bằng `<?` và kết thúc bằng `?>`.
- Các thẻ do người dùng tự định nghĩa chẳng hạn như `<DOCUMENT>`, `<GREETING>`, `<MESSAGE>`.
- Thẻ luôn bắt đầu bằng `<` và kết thúc bằng `>`.
- Phải có thẻ mở và đóng duy nhất cho toàn bộ tài liệu (root).

Hiển thị tài liệu XML

- Trình duyệt chỉ có thể hiện thị file XML bằng cách dữ toàn bộ nội dung file XML lên màn hình.



Định kiểu XML

- Định dạng file XML bằng CSS(Stylesheet).
- Định dạng file XML bằng XSLT.
- CSS và XSLT dùng để định kiểu và biến đổi XML để hiển thị dữ liệu phía người dùng không khác gì HTML.
- Dùng DOM, SAX để rút trích dữ liệu từ file XML kết hợp với các thẻ định dạng của HTML để hiển thị phía người dùng.

Định kiểu XML

- Ví dụ: với file XML như ví dụ trước, kết hợp với XSLT để định kiểu như sau:

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns="http://www.w3.org/TR/REC-html40">
    <xsl:template match="/">
        <html>
            <head>
                <title> <xsl:value-of select="DOCUMENT/GREETING"/> </title>
            </head>
            <body>
                <h1 align="center"><font color="#008080"> <xsl:value-of
                    select="DOCUMENT/MESSAGE" /> </font></h1>
            </body>
        </html>
    </xsl:template>
</xsl:stylesheet>
```

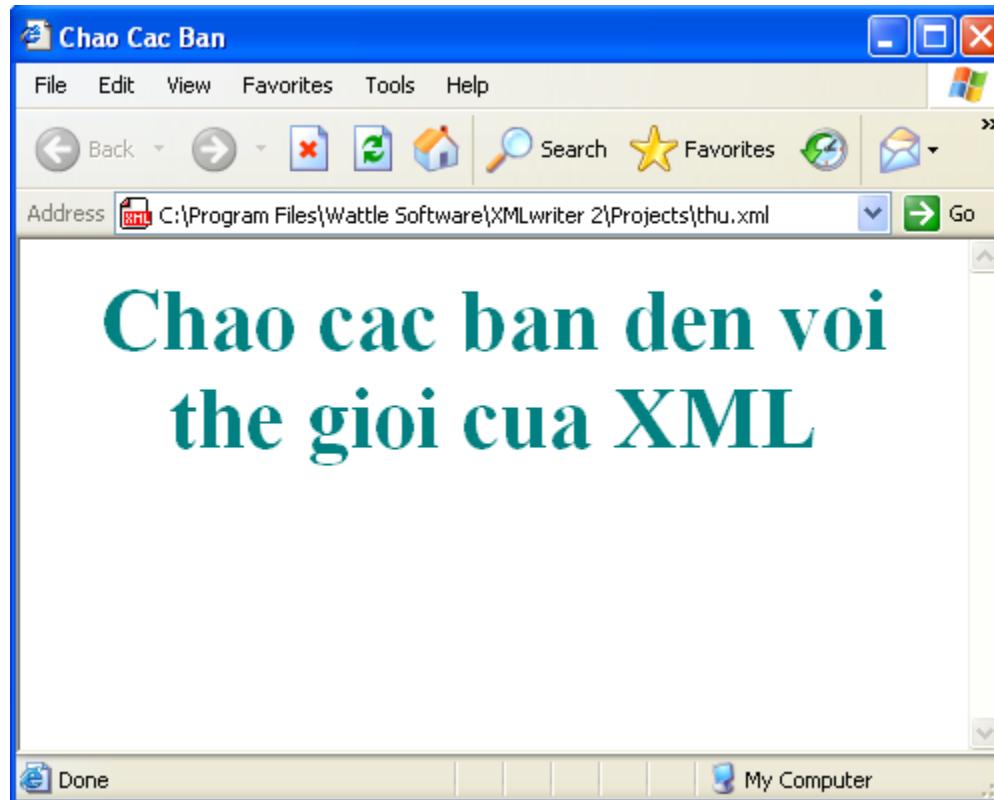
Định kiểu XML

Ví dụ(tt):

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="thu.xsl" ?>
<DOCUMENT>
    <GREETING>
        Chao Cac Ban
    </GREETING>
    <MESSAGE>
        Chao cac ban den voi the gioi cua XML
    </MESSAGE>
</DOCUMENT>
```

Định kiểu XML

- Hiển thị tài liệu XML trong trình duyệt IE



Trình duyệt XML

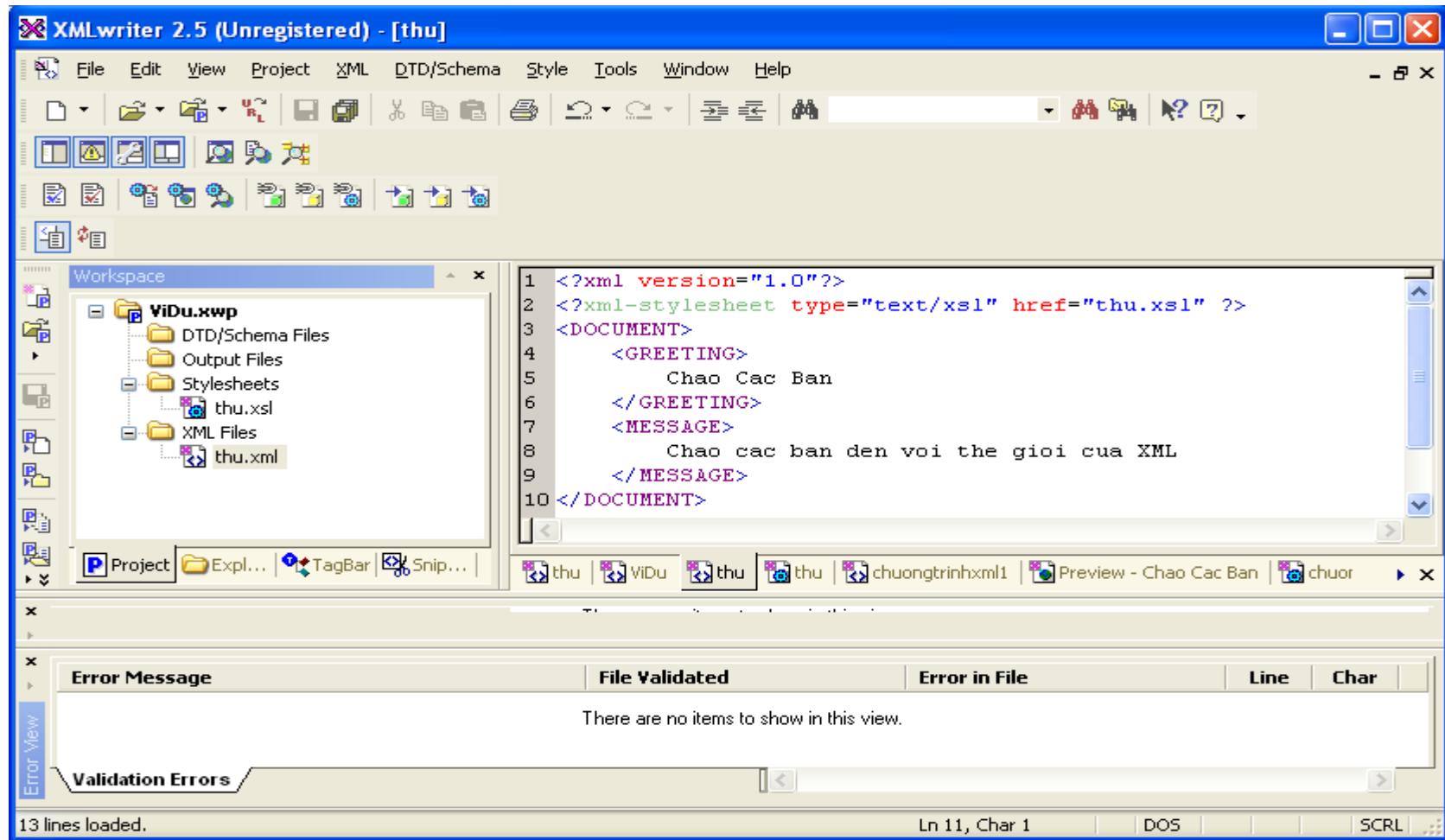
■ Internet Explorer (IE)

- Vẫn là trình duyệt XML mạnh nhất hiện nay.
- Cho phép dùng Javascript để lập trình và truy xuất dữ liệu XML.
- Ngoài ra còn hỗ trợ cả Jscript, Vbscript.
- Hỗ trợ CSS.

■ Netscape Navigation 6

- Phiên bản 6.0 hỗ trợ XML khá chuẩn.
- Hỗ trợ CSS
- Hỗ trợ các ngôn ngữ XML mở rộng.

Trình soạn thảo XMLWriter



Ứng dụng XML

- XML có thể tạo ra tập các ngôn ngữ con khác.
- Ứng dụng XML mang ý nghĩa cho biết một tập các thẻ hay tập con XML hoạt động riêng trong một lĩnh vực nào đó.
- MathML: định dạng các biểu thức, kí hiệu toán học.
- CML: Ngôn ngữ định dạng hóa học.

Ứng dụng XML

- CDF: khuôn dạng định nghĩa kênh(Channel Definition Format).
- SMIL: ngôn ngữ tích hợp multimedia đồng bộ.
- XHTML: dùng mở rộng và định nghĩa lại ngôn ngữ định dạng HTML.
- XUL: ngôn ngữ cấu hình giao diện người dùng

Ứng dụng XML

- VML: ngôn ngữ định dạng Vector(Vector markyp Language).
- WML: Ngôn ngữ định dạng mạng không dây.
- SOAP: Giao thức truy cập đối tượng đơn giản(Simple Object Access Protocol)

Kết luận

- Tổng quan về XML
- Ứng dụng mà XML đạt được hiện nay.

Lược đồ XML (XML Schema)

Mục tiêu

- Đọc và tạo XML Schema
- Làm thế nào để sử dụng được XML Schema trong ứng dụng.
- Vì sao XML Schema mạnh hơn DTD

Giới thiệu

- Vì sự phức tạp của khai báo DTD, tổ chức W3C đưa ra một giải pháp tổng quát hơn DTD đó là khai báo và định nghĩa các phần tử trong tài liệu XML theo lược đồ XML (XML Schema).
- Để kiểm tra tính hợp lệ của tài liệu XML bằng lược đồ XML Schema, ta dùng các bộ kiểm tra cú pháp lược đồ (Schema Checker).

Giới thiệu

- Định nghĩa những phần tử xuất hiện trong tài liệu XML.
- Định nghĩa những thuộc tính xuất hiện trong tài liệu.
- Định nghĩa quan hệ phần tử cha con
- Định nghĩa thứ tự các phần tử con
- Định nghĩa số phần tử con
- Định nghĩa phần tử rỗng hay chứa dữ liệu text
- Định nghĩa kiểu dữ liệu của phần tử và thuộc tính
- Định nghĩa giá trị mặc định của thuộc tính và phần tử

Tại sao XML sử dụng XML Schema?

- Dễ dàng để mô tả nội dung tài liệu vì dùng chính cú pháp XML để định nghĩa
- Dễ kiểm tra tính hợp lệ của tài liệu
- Dễ định nghĩa về mặt dữ liệu (data facet)
- Dễ dàng định nghĩa dữ liệu mẫu (data patterns)
- Dễ chuyển đổi kiểu dữ liệu này sang kiểu dữ liệu khác

Ví dụ

■ Note.xml

```
<?xml version="1.0"?>  
<note>  
  <to>Tove</to>  
  <from>Jani</from>  
  <heading>Reminder</heading>  
  <body>Don't forget me this weekend!</body>  
</note>
```

Ví dụ (tt)

■ Note.dtd

```
<!ELEMENT note (to, from, heading, body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
```

Ví dụ (tt)

- Note.xsd

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.w3schools.com"
xmlns="http://www.w3schools.com" elementFormDefault="qualified">
  <xs:element name="note">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="to" type="xs:string"/>
        <xs:element name="from" type="xs:string"/>
        <xs:element name="heading" type="xs:string"/>
        <xs:element name="body" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Ví dụ (tt)

- Tài liệu XML có tham chiếu file DTD

```
<?xml version="1.0"?>
<!DOCTYPE note SYSTEM
"http://www.w3schools.com/dtd/note.dtd">
<note>
    <to>Tove</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>
</note>
```

Ví dụ (tt)

- Tài liệu XML tham chiếu lược đồ XML

```
<?xml version="1.0"?>
<note xmlns="http://www.w3schools.com"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.w3schools.com note.xsd">
    <to>Tove</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>
</note>
```

Lược đồ XML – Phần tử <schema>

- <schema> là phần tử gốc của mọi lược đồ XML Schema.

```
<?xml version="1.0"?>  
<xs:schema>
```

...

...

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

- Trong lược đồ XML Schema chứa một vài thuộc tính như sau:

```
<?xml version="1.0"?>  
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"  
targetNamespace="http://www.w3schools.com"  
xmlns="http://www.w3schools.com"  
elementFormDefault="qualified">
```

...

...

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

Lược đồ XML – Phần tử <schema>

- xmlns:xs="http://www.w3.org/2001/XMLSchema" chỉ ra các phần tử và kiểu dữ liệu dùng trong lược đồ từ http://www.w3.org/2001/XMLSchema. Chỉ định này bảo cho bộ kiểm tra cú pháp lược đồ rằng tất cả các phần tử dùng trong tài liệu XML đều được khai báo trong namespace "http://www.w3schools.com".
- xsi:schemaLocation="http://www.w3schools.com note.xsd"

Định nghĩa phần tử đơn

- Phần tử đơn là phần tử chỉ chứa dữ liệu text, không chứa các phần tử khác hay thuộc tính.
- Kiểu text trong XML Schema có thể là kiểu boolean, string, date...
- Cú pháp để định nghĩa một phần tử đơn:
`<xss:element name="xxx" type="yyy"/>`

Định nghĩa phần tử đơn

- Trong đó xxx là tên của phần tử và yyy là kiểu dữ liệu của phần tử.
- XML schema đã xây dựng sẵn nhiều kiểu dữ liệu. Một vài kiểu dữ liệu phổ biến:
 - xs:string
 - xs:decimal
 - xs:integer
 - xs:boolean
 - xs:date
 - xs:time

Ví dụ

- Trong tài liệu XML có các phần tử sau:

```
<lastname>Refsnes</lastname> <age>36</age>  
<dateborn>1970-03-27</dateborn>
```

- Định nghĩa trong XML Schema như sau:

```
<xss:element name="lastname" type="xs:string"/>  
<xss:element name="age" type="xs:integer"/>  
<xss:element name="dateborn" type="xs:date"/>
```

Đặt giá trị mặc định cho phần tử đơn

■ Giá trị mặc định:

- Dùng Default: phần tử sẽ được tự động gán giá trị mặc định nếu nó không được gán bởi giá trị khác được.
- Dùng Fixed: giá trị của phần tử sẽ được gán bằng giá trị mặc định và không thay đổi.

■ Ví dụ:

```
<xs:element name="color" type="xs:string" default="red"/>  
<xs:element name="color" type="xs:string" fixed="red"/>
```

Tạo các kiểu dữ liệu đơn giản

- Restriction: dùng để định nghĩa các giá trị cho phần tử hay thuộc tính trong tài liệu XML.
- Restriction trên giá trị:

Ví dụ: muốn định nghĩa một phần tử tên là AGE và giá trị của nó chỉ nằm từ 0 đến 120.

```
<xs:element name="age">
  <xs:simpleType>
    <xs:restriction base="xs:integer">
      <xs:minInclusive value="0"/>
      <xs:maxInclusive value="120"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Tạo các kiểu dữ liệu đơn giản

- Restriction trên một tập các giá trị:

Ví dụ: định nghĩa một phần tử CAR mà giá trị của nó nằm trong tập các giá trị sau: BMW, TOYOTA, FORD

```
<xs:element name="CAR">  
  <xs:simpleType>  
    <xs:restriction base="xs:string">  
      <xs:enumeration value="BMW"/>  
      <xs:enumeration value="TOYOTA"/>  
      <xs:enumeration value="FORD"/>  
    </xs:restriction>  
  </xs:simpleType>  
</xs:element>
```

Tạo các kiểu dữ liệu đơn giản

- Restriction trên một tập các giá trị:

Với ví dụ trên có cách viết tương tự:

```
<xs:element name="CAR" type="carType"/>
<xs:simpleType name="carType">
    <xs:restriction base="xs:string">
        <xs:enumeration value="BMW"/>
        <xs:enumeration value="TOYOTA"/>
        <xs:enumeration value="FORD"/>
    </xs:restriction>
</xs:simpleType>
```

Tạo các kiểu dữ liệu đơn giản

- Restriction trên một chuỗi các giá trị:

Ví dụ: định nghĩa một phần tử “LETTER” mà giá trị của nó chỉ là một trong các ký tự thường từ a đến z:

```
<xs:element name="LETTER">  
  <xs:simpleType>  
    <xs:restriction base="xs:string">  
      <xs:pattern value="[a-z]"/>  
    </xs:restriction>  
  </xs:simpleType>  
</xs:element>
```

Tạo các kiểu dữ liệu đơn giản

- Restriction trên một chuỗi các giá trị:

Ví dụ: định nghĩa một phần tử “initials” mà giá trị của nó là 3 kí tự chữ hoa a đến z:

```
<xs:element name="initials">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[A-Z][A-Z][A-Z]"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Tạo các kiểu dữ liệu đơn giản

- Restriction trên một chuỗi các giá trị:

Ví dụ: định nghĩa một phần tử “initials” mà giá trị của nó là 3 kí tự chữ hoa a đến z hoặc chữ thường từ a đến z:

```
<xs:element name="initials">  
  <xs:simpleType>  
    <xs:restriction base="xs:string">  
      <xs:pattern value="[a-zA-Z][a-zA-Z][a-zA-Z]"/>  
    </xs:restriction>  
  </xs:simpleType>  
</xs:element>
```

Tạo các kiểu dữ liệu đơn giản

- Restriction trên một chuỗi các giá trị:

Ví dụ: định nghĩa một phần tử “choice” mà giá trị của nó là 3 ký tự chữ thường từ x, y hoặc z:

```
<xs:element name="choice">  
  <xs:simpleType>  
    <xs:restriction base="xs:string">  
      <xs:pattern value="[xyz]"/>  
    </xs:restriction>  
  </xs:simpleType>  
</xs:element>
```

Tạo các kiểu dữ liệu đơn giản

- Restriction trên một chuỗi các giá trị:

Ví dụ: định nghĩa một phần tử “prodid” mà giá trị của nó là một số 5 chữ số từ 0 đến 9:

```
<xs:element name="choice">
  <xs:simpleType>
    <xs:restriction base="xs:integer">
      <xs:pattern value="[0-9][0-9][0-9][0-9][0-9]"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Tạo các kiểu dữ liệu đơn giản

- Restriction trên một chuỗi các giá trị:

Ví dụ: định nghĩa một phần tử “letter” mà giá trị của nó là không có hoặc là một chuỗi gồm nhiều kí tự thường từ a-z:

```
<xs:element name="LETTER">  
  <xs:simpleType>  
    <xs:restriction base="xs:string">  
      <xs:pattern value="([a-z])*/>  
    </xs:restriction>  
  </xs:simpleType>  
</xs:element>
```

Tạo các kiểu dữ liệu đơn giản

- Restriction trên một chuỗi các giá trị:

Ví dụ: định nghĩa một phần tử “letter” mà giá trị của nó là một hoặc nhiều cặp kí tự mà mỗi cặp là một kí tự thường kèm theo một kí tự hoa sau nó: (sToP, không được là STOP hay StOp):

```
<xs:element name="LETTER">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:pattern value="([a-z][A-Z])+" />
        </xs:restriction>
    </xs:simpleType>
</xs:element>
```

Tạo các kiểu dữ liệu đơn giản

- Restriction trên một chuỗi các giá trị:

Ví dụ: định nghĩa một phần tử “gender” mà giá trị của nó là một trong hai giá trị male hoặc female:

```
<xs:element name="gender">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="male|female"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Tạo các kiểu dữ liệu đơn giản

- Restriction trên một chuỗi các giá trị:

Ví dụ: định nghĩa một phần tử “password” mà giá trị của nó bắt buộc phải là 8 ký tự, các ký tự có thể là hoa hoặc thường từ a đến z và chữ số từ 0 đến 9:

```
<xs:element name="password">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[a-zA-Z0-9] {8}" />
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Tạo các kiểu dữ liệu đơn giản

- Restriction trên chiều dài:

Ví dụ: định nghĩa một phần tử “password” mà giá trị của nó là một chuỗi 8 kí tự

```
<xs:element name="LETTER">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:length value="8" />
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Tạo các kiểu dữ liệu đơn giản

- Restriction trên chiều dài:

Ví dụ: định nghĩa một phần tử “password” mà giá trị của nó là một chuỗi ≥ 5 và ≤ 8 kí tự:

```
<xs:element name="password">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:minLength value="5"/>
      <xs:maxLength value="8"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Bài tập

■ Bài 1:

- ❑ Một chương trình quản lý thông tin sinh viên, và điểm các môn học mà họ đăng ký lưu các thông tin sau: sinh viên gồm mã số sinh viên, họ tên sinh viên, lớp. Một môn học mà sinh viên đăng ký học gồm có thông tin mã môn học, tên môn học, số tín chỉ, với mỗi môn học mà sinh viên đăng ký học thì sẽ có kết quả cuối kì chính là điểm thi mà sinh viên đạt được trong môn đó.
- ❑ Yêu cầu: định nghĩa lược đồ XML Schema với yêu cầu như sau:

Bài tập

- ❑ MSSV: là một chuỗi các kí tự số có chiều dài 7 kí tự.
- ❑ Lớp của sinh viên là một trong những giá trị của tập hợp gồm các lớp {CTK28, CTK28CD, CTK29, CTK29CD, CTK30, CTK30CD, CTK31, CTK31CD}.
- ❑ Mã môn học là một chuỗi 4 kí tự, hai kí tự đầu là chữ cái, hai kí tự sau là số.
- ❑ Số tín chỉ của môn học ≥ 1 và ≤ 5 .
- ❑ Điểm của sinh viên ≥ 0 và ≤ 10

Bài tập

■ Bài 4

- ❑ Mỗi đơn hàng, người ta cần lưu các thông tin sau: Mã khách hàng, tên khách hàng, địa chỉ liên lạc và một danh sách những mặt hàng người đó mua. Danh sách mặt hàng gồm có nhiều mặt hàng khác nhau, mỗi mặt hàng gồm những thông tin sau: Mã mặt hàng, tên mặt hàng, số lượng, đơn giá.
- ❑ Định nghĩa XML Schema với yêu cầu như sau:

Bài tập

- ❑ MaKH và MaMH: là một chuỗi 4 kí tự trong đó hai kí tự đầu là chữ cái, hai kí tự sau là kí số.
- ❑ DiaChi: là một chuỗi các kí tự với định dạng như sau: bắt đầu phải là số nhà, sau đến tên đường.
- ❑ Số lượng và đơn giá là kiểu số và phải là số dương.

Bài tập

■ Bài 5:

- Cho cấu trúc XML lưu trữ thông tin những cuốn sách đã được xuất bản theo từng lĩnh vực. Mỗi lĩnh vực có tên và có thể cha có sách xuất bản hoặc cũng có thể đã có nhiều cuốn. Mỗi cuốn sách có thông tin một tựa đề duy nhất, một hay nhiều tác giả, mỗi tác giả lại có thông tin mã tác giả, tên tác giả, địa chỉ, số điện thoại với mã tác giả, tên tác giả là duy nhất cho mỗi người, địa chỉ email, số điện thoại có thể không có hoặc chỉ có một thông tin độc nhất cho mỗi người
- Hãy định nghĩa XML Schema.

Bài tập

- ❑ Mã sách và mã tác giả là một chuỗi gồm 4 kí tự trong đó hai kí tự đầu là chữ và hai kí tự sau là số.
- ❑ Địa chỉ email là một chuỗi với định dạng ##@##.##.
- ❑ Số điện thoại là một chuỗi các kí số với định dạng sau: MaVung.SoĐT

Định nghĩa thuộc tính

■ Một phần tử đơn giản thì không có thuộc tính.

■ Cú pháp định nghĩa thuộc tính:

```
<xs:attribute name="xxx" type="yyy"/>
```

■ Trong đó:

- ❑ xxx là tên thuộc tính

- ❑yyy là kiểu dữ liệu của thuộc tính

■ Ví dụ: với phần tử có thuộc tính như sau:

<lastname lang="EN">Smith</lastname> thì định nghĩa thuộc tính như sau:

```
<xs:attribute name="lang" type="xs:string"/>
```

Định nghĩa thuộc tính

- Các kiểu dữ liệu của thuộc tính:
 - ❑ xs:string
 - ❑ xs:decimal
 - ❑ xs:integer
 - ❑ xs:boolean
 - ❑ xs:date
 - ❑ xs:time
- Chỉ định ràng buộc và trị mặc định cho thuộc tính:
 - ❑ Required: yêu cầu thuộc tính phải có và được gán giá trị trong thẻ
 - ❑ Optional: thuộc tính có thể có hoặc không
 - ❑ Fixed: giá trị thuộc tính phải được gán cố định và không thay đổi.
 - ❑ Default: nếu thuộc tính không được gán giá trị thì giá trị mặc định trong lược đồ sẽ là giá trị của thuộc tính.
- Sử dụng từ khóa use để chỉ định ràng buộc cho thuộc tính.

Ví dụ

```
<xs:element name="DOCUMENT">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Address" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Name" type="xs:string" minOccurs="0" />
            <xs:element name="Street" type="xs:string" minOccurs="0" />
            <xs:element name="City" type="xs:string" minOccurs="0" />
          </xs:sequence>
          <xs:attribute name="Phone" type="xs:string" use="optional" />
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Định nghĩa phần tử phức tạp

- Phần tử phức tạp là phần tử chứa các phần tử khác hay các thuộc tính.
- Có 4 loại phần tử phức tạp:
 - Những phần tử rỗng
 - Phần tử chứa các phần tử khác
 - Phần tử chứa dữ liệu text
 - Phần tử chứa cả dữ liệu text và cả các phần tử
 - Mỗi loại phần tử này có thể chứa nhiều thuộc tính

Một ví dụ phần tử phức

- Ví dụ phần tử rỗng:

```
<Product pid="1234">
```
- Phần tử Employee chứa các phần tử khác

```
<employee>
    <firstname>John</firstname>
    <lastname>Smith</lastname>
</employee>
```
- Phần tử Food chỉ chứa kiểu dữ liệu text

```
<food type="dessert">Ice cream</food>
```
- Phần tử description chứa cả text và phần tử khác

```
<description>It happened on <date lang="norwegian">03.03.99</date> ....
</description>
```

Định nghĩa phần tử phức tạp

- Ví dụ, với phần tử employee

```
<employee>
  <firstname>John</firstname>
  <lastname>Smith</lastname>
</employee>
```

ta có định nghĩa như sau:

```
<xsd:element name="employee">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="firstname" type="xsd:string"/>
      <xsd:element name="lastname" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

Định nghĩa phần tử phức tạp

- Ví dụ, với phần tử employee

```
<employee>
    <firstname>John</firstname>
    <lastname>Smith</lastname>
</employee>
```

ta có thể dùng thuộc tính để tham chiếu đến phần tử khác:

```
<xs:element name="employee" type="personinfo"/>
<xs:element name="student" type="personinfo"/>
<xs:element name="member" type="personinfo"/>
<xs:complexType name="personinfo">
    <xs:sequence>
        <xs:element name="firstname" type="xs:string"/>
        <xs:element name="lastname" type="xs:string"/>
    </xs:sequence>
</xs:complexType>
</xs:element>
```

Định nghĩa phần tử phức tạp

- Ta cũng có thể định nghĩa một phần tử phức tạp dựa trên cơ sở một phần tử phức tạp khác đã có và thêm vào một số phần tử.

```
<xs:element name="employee" type="fullpersoninfo"/>
<xs:complexType name="personinfo">
    <xs:sequence>
        <xs:element name="firstname" type="xs:string"/>
        <xs:element name="lastname" type="xs:string"/>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="fullpersoninfo">
    <xs:complexContent>
        <xs:extension base="personinfo">
            <xs:sequence>
                <xs:element name="address" type="xs:string"/>
                <xs:element name="city" type="xs:string"/>
                <xs:element name="country" type="xs:string"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

Định nghĩa phần tử phức tạp

- Phần tử rỗng:

Ví dụ 1: <product prodid="1345" />

Ví dụ 2:

```
<xs:element name="product">
    <xs:complexType>
        <xs:attribute name="prodid"
                      type="xs:positiveInteger"/>
    </xs:complexType>
```

```
</xs:element>
```

tương tự

```
<xs:element name="product" type="prodtype"/>
<xs:complexType name="prodtype">
    <xs:attribute name="prodid"
                  type="xs:positiveInteger"/>
</xs:complexType>
```

Định nghĩa phần tử phức tạp

- Phần tử chứa các phần tử khác:

Ví dụ: để định nghĩa phần tử person như sau:

```
<person>
    <firstname>John</firstname>
    <lastname>Smith</lastname>
</person>
```

ta định nghĩa như sau:

```
<xsd:element name="person">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="firstname" type="xsd:string"/>
            <xsd:element name="lastname" type="xsd:string"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
```

Chú ý: thẻ `<xsd:sequence>` để đặt thứ tự xuất hiện của các phần tử trong tài liệu.

Định nghĩa phần tử phức tạp

- Định nghĩa phần tử chứa dữ liệu text

Ví dụ:

```
<xs:element name="somename">
    <xs:complexType>
        <xs:simpleContent>
            <xs:extension base="basetype">
                ....
                ....
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
</xs:element> hoặc
<xs:element name="somename">
    <xs:complexType>
        <xs:simpleContent>
            <xs:restriction base="basetype">
                ....
                ....
            </xs:restriction>
        </xs:simpleContent>
    </xs:complexType>
</xs:element>
```

Định nghĩa phần tử phức tạp

- Định nghĩa phần tử chứa dữ liệu hỗn hợp

Ví dụ: ta định nghĩa phần tử như sau:

```
<letter>
```

 Dear Mr.<name>John Smith</name>. Your order

 <orderid>1032</orderid> will be shipped on <shipdate>2001-07-13</shipdate>.

```
</letter>
```

Khi đó ta định nghĩa như sau:

```
<xsd:element name="letter">
    <xsd:complexType mixed="true">
```

```
        <xsd:sequence>
```

```
            <xsd:element name="name" type="xsd:string"/>
```

```
            <xsd:element name="orderid" type="xsd:positiveInteger"/>
```

```
            <xsd:element name="shipdate" type="xsd:date"/>
```

```
        </xsd:sequence>
```

```
    </xsd:complexType>
```

```
</xsd:element>
```

Một số chỉ định

- Thứ tự:
 - All: các phần tử con có thể xuất hiện theo thứ tự bất kì

```
<xs:element name="person">
<xs:complexType>
<xs:all>
<xs:element name="firstname" type="xs:string"/>
<xs:element name="lastname" type="xs:string"/>
</xs:all>
</xs:complexType>
</xs:element>
```
 - Choice: chỉ định hoặc điều này xảy ra hoặc điều khác xảy ra

```
<xs:element name="person">
<xs:complexType>
<xs:choice>
<xs:element name="employee" type="employee"/>
<xs:element name="member" type="member"/>
</xs:choice> </xs:complexType></xs:element>
```

Một số chỉ định

- Thứ tự:
 - Sequence: yêu cầu các phần tử con xuất hiện theo đúng thứ tự
- Chỉ định số lần:
 - maxoccur

```
<xss:element name="person">
<xss:complexType>
    <xss:sequence>
        <xss:element name="full_name" type="xs:string"/>
        <xss:element name="child_name" type="xs:string" maxOccurs="10"/>
    </xss:sequence>
</xss:complexType>
</xss:element>
```

ở ví dụ trên chỉ định rằng child_name xuất hiện ít nhất 1 lần và nhiều nhất 10 lần
 - minoccur

Một số chỉ định

- Chỉ định nhóm: có thể định nghĩa và gom các phần tử lại thành một nhóm. Nhóm giống như biểu thức () trong DTD
- Để định nghĩa một nhóm ta dùng cú pháp sau:

```
<xs:group name="groupname">  
...  
</xs:group>
```
- Ví dụ muốn độc giả có thể vừa mượn sách vừa mượn tạp chí, ta gom phần tử books và magazines vào thành một nhóm

```
<xs:group name="booksandmagazines">  
    <xs:element ref="books">  
    <xs:element ref="magazines">  
</xs:group>
```

Một số chỉ định

- Ta có thể kết hợp all (các phần tử trong nhóm phải xuất hiện đồng thời trong tài liệu hoặc không xuất hiện lần nào cả), choice, sequence khi định nghĩa nhóm

```
<xs:group name="persongroup">  
    <xs:sequence>  
        <xs:element name="firstname" type="xs:string"/>  
        <xs:element name="lastname" type="xs:string"/>  
        <xs:element name="birthday" type="xs:date"/>  
    </xs:sequence>  
</xs:group>
```

Một số chỉ định

- Ta có thể định nghĩa một nhóm, và định nghĩa nhóm khác tham chiếu đến nó

```
<xs:group name="persongroup">
    <xs:sequence>
        <xs:element name="firstname" type="xs:string"/>
        <xs:element name="lastname" type="xs:string"/>
        <xs:element name="birthday" type="xs:date"/>
    </xs:sequence>
</xs:group>
<xs:element name="person" type="personinfo"/>
<xs:complexType name="personinfo">
    <xs:sequence>
        <xs:group ref="persongroup"/>
        <xs:element name="country" type="xs:string"/>
    </xs:sequence>
</xs:complexType>
```

Không gian tên miền (Namespace)

■ Thuộc tính

xsi:noNamespaceSchemaLocation

```
<xsi:noNamespaceSchemaLocation="anyURI" >
```

- Khi trong lược đồ XMLSchema không yêu cầu namespace. Để chỉ định vị trí cho lược đồ XMLSchema không dùng target namespace.

Không gian tên miền (Namespace)

- Ví dụ: sinhvien.xsd

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema id="SinhVien"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="ThongTinSV">
<xs:complexType>
<xs:sequence>
<xs:element name="SinhVien" maxOccurs="unbounded">
<xs:complexType>
<xs:sequence>
<xs:element name="MSSV" type="xs:string"/>
<xs:element name="HoTen" type="xs:string" />
```

Không gian tên miền (Namespace)

- Ví dụ(tt): sinhvien.xml

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="sinhvien.xsl"?>
<ThongTinSV xmlns:xs="http://www.w3.org/2001/XMLSchema-instance" xs:noNamespaceSchemaLocation="SinhVien1.xsd">
    <SinhVien>
        <MSSV>0413213</MSSV>
        <HoTen>Nguyen Quang Dung</HoTen>
    </SinhVien>
    ...
<ThongTinSV>
```

Không gian tên miền (Namespace)

■ Thuộc tính **xsi:SchemaLocation**

```
<xsi:schemaLocation="list of anyURI" >
```

- Thuộc tính này được dùng trong tài liệu XMLSchema có target namespace. Danh sách anyURI mỗi phần tử là một cặp gồm namesapce và mô tả vị trí chỉ định.

Không gian tên miền (Namespace)

- Ví dụ: sinhvien.xsd

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema id="SinhVien" targetNamespace="http://sinhvien"
elementFormDefault="qualified" xmlns:sv="http://sinhvien"
xmlns:mstns="http://sinhvien/SinhVien.xsd"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:include schemaLocation="SinhVien.xsd"/>
<xs:element name="ThongTinSV">
<xs:complexType>
<xs:sequence>
<xs:element name="SinhVien" maxOccurs="unbounded">
<xs:complexType>
<xs:sequence>
<xs:element name="MSSV" type="xs:string"/>
<xs:element name="HoTen" type="xs:string" />
```

Không gian tên miền (Namespace)

- Ví dụ(tt): sinhvien.xml

```
<sv:ThongTinSV  
xmlns:xs="http://www.w3.org/2001/XMLSchema-  
instance" xs:schemaLocation="http://sinhvien  
SinhVien.xsd" xmlns:sv="http://sinhvien" >  
  
<sv:SinhVien>  
    <sv:MSSV>0413113</sv:MSSV>  
    <sv:HoTen>Nguyen Van Hung</sv:HoTen>  
  </sv:SinhVien>  
</sv:ThongTinSV>
```

CSS

(Cascading Style sheet)

Giới thiệu

- CSS viết tắt của Cascading Style Sheet.
- Chỉ định cách hiển thị của các phần tử HTML.
- CSS là một sự đột phá trong thiết kế web bởi vì:
 - Nó cho phép người phát triển ứng dụng web kiểm soát được kiểu và cách bố trí ở nhiều trang web một lúc.
 - Định nghĩa một lần và áp dụng cho tất cả các trang khác.
 - Sửa một lần và tất cả các trang tự động cập nhật thay đổi

Cú pháp CSS

- selector {property: value}
- Trong đó:
 - selector là tên thẻ hay phần tử muốn định nghĩa.
 - Property là thuộc tính bạn muốn thay đổi.
 - Value: là giá trị thuộc tính. Nếu giá trị có nhiều từ thì phải đặt trong dấu nháy kép
- Ví dụ:

```
body {color: black}  
p {font-family: "sans serif"}
```

Cú pháp CSS

- Nếu có nhiều thuộc tính muốn định nghĩa thì phải phân cách nhau bằng dấu ;
Ví dụ: p {text-align:center;color:red}
- Để định nghĩa dễ đọc có thể đặt các thuộc tính định nghĩa ở các dòng khác nhau
Ví dụ:

```
P
{
    text-align: center;
    color: black;
    font-family: arial
}
```

Cú pháp CSS

- Chúng ta có thể định nghĩa một nhóm các phần tử và phân cách giữa các phần tử bằng dấu,

Ví dụ:

h1,h2,h3,h4,h5,h6

{

 color: green

}

- Định nghĩa lớp: dùng để định nghĩa kiểu khác nhau cho cùng một thẻ

Ví dụ:

p.right {text-align: right}

p.center {text-align: center}

Trong tài liệu HTML sử dụng thuộc tính như sau:

<p class="right">This paragraph will be right-aligned.</p>

<p class="center">This paragraph will be center-aligned.</p>

Cú pháp CSS

- Ta có thể áp dụng một hoặc nhiều lớp vào trong một phần tử
Ví dụ:

```
<p class="center bold">  
    This is a paragraph.  
</p>
```

→ lớp center và lớp bold cùng được áp dụng cho phần tử p
- Chúng ta cũng có thể định nghĩa lớp dùng cho mọi phần tử
 - Cú pháp: .tenlop {các định nghĩa}
 - Ví dụ:
.center {text-align: center}
áp dụng lớp center vào cho các thẻ, ví dụ

```
<h1 class="center">  
    This heading will be center-aligned  
</h1>  
<p class="center">  
    This paragraph will also be center-aligned.  
</p>
```

Cú pháp CSS

- Có thể áp dụng kiểu cho phần tử có giá trị thuộc tính thỏa một giá trị cụ thể nào đó.

Ví dụ:

```
Input[type="text"] {background-color: blue}
```

ví dụ trên áp dụng kiểu cho phần tử Input với giá trị thuộc tính type phải bằng “text”

- Có thể định nghĩa kiểu cho phần tử bằng định danh ví dụ:

```
#green {color: green}
```

```
p#para1{text-align: center;color: red}
```

Cú pháp CSS

- Chú thích: dùng dấu /* và */ để chú thích trong CSS.

```
/* This is a comment */
```

```
p  
{
```

```
    text-align: center;  
    /* This is another comment */  
    color: black;  
    font-family: arial
```

```
}
```

Sử dụng CSS như thế nào?

■ Ví dụ1:

```
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="ex1.css" />
  </head>
  <body>
    <h1>This header is 36 pt</h1>
    <h2>This header is blue</h2>
    <p>This paragraph has a left margin of 50 pixels</p>
  </body>
</html>
```

Sử dụng CSS như thế nào?

- Ví dụ 1(tt): file ex1.css

```
body { background-color: yellow }  
h1 { font-size: 36pt }  
h2 { color: blue }  
p { margin-left: 50px }
```

- Hiển thị:

This header is 36 pt

This header is blue

This paragraph has a left margin of 50 pixels

Sử dụng CSS như thế nào

- Ví dụ 2 (tt):

```
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="ex2.css" />
  </head>
  <body>
    <h1>This is a header 1</h1>
    <hr />
    <p>You can see that the style sheet formats the text</p>
    <p>
      <a href="http://www.w3schools.com" target="_blank">This is a
      link</a>
    </p>
  </body>
</html>
```

Sử dụng CSS như thế nào?

- Ví dụ 2 (tt): file ex2.css

```
body {background-color: tan}  
h1 {color:maroon; font-size:20pt}  
hr {color:navy}  
p {font-size:11pt; margin-left: 15px}  
a:link {color:green}  
a:visited {color:yellow}  
a:hover {color:black}  
a:active {color:blue}
```

- Hiển thị:

This is a header 1

You can see that the style sheet formats the text

This is a link

Sử dụng CSS như thế nào?

- Khi trình duyệt đọc một sheet style, nó sẽ định dạng tài liệu theo style sheet đó. Có 3 cách để đưa style sheet vào tài liệu:

- Style sheet ngoại: dùng khi muốn áp dụng kiểu cho nhiều trang. Mỗi trang muốn link đến style sheet thì phải dùng thẻ `<link>`

Ví dụ:

```
<head>
  <link rel="stylesheet" type="text/css"
        href="mystyle.css" />
</head>
```

Sử dụng CSS như thế nào?

- Style sheet nội: nên sử dụng khi chỉ kiểu chỉ dùng cho một trang riêng. Định nghĩa style sheet nội trong thẻ `<style>`

Ví dụ:

```
<head>
<style type="text/css">
    hr {color: sienna}
    p {margin-left: 20px}
    body {background-image: url("images/back40.gif")}
</style>
</head>
```

Sử dụng CSS như thế nào?

- **Inline style:** dùng inline style sẽ mất một số lợi ích của style sheet bởi vì nó kết hợp nội dung với trình diễn.

```
<p style="color: sienna; margin-left: 20px">This is a paragraph</p>
```

- **Multiple style sheet:** có một vài thuộc tính được đặt cùng tên nhưng khác style sheet, giá trị sẽ kế thừa từ nhiều style sheet chỉ định.

style sheet ngoại định nghĩa h3 như sau:

```
h3
{
    color: red;
    text-align: left;
    font-size: 8pt
}
```

và style sheet nội định nghĩa h3 như sau:

```
h3
{
    text-align: right;
    font-size: 20pt
}
```

Khi đó trang có style sheet nội trên nếu dùng style sheet ngoại thì kết quả là

```
color: red;
text-align: right;
font-size: 20pt
```

CSS Background

- Thuộc tính CSS background cho phép bạn điều chỉnh được màu nền của một phần tử, cho phép đặt hình làm nền, đặt lặp lại nhiều lần hình theo chiều ngang hay dọc để làm nền, đặt vị trí của hình trong trang.

Thuộc tính	Mô tả	Giá trị
<u>Background</u>	A shorthand property for setting all background properties in one declaration	<i>background-color</i> <i>background-image</i> <i>background-repeat</i> <i>background-attachment</i> <i>background-position</i>
<u>background-repeat</u>	Sets if/how a background image will be repeated	repeat repeat-x repeat-y no-repeat
<u>Background-attachment</u>	Sets whether a background image is fixed or scrolls with the rest of the page	scroll fixed

CSS BackGround

<u>background-color</u>	Sets the background color of an element	<i>color-rgb</i> <i>color-hex</i> <i>color-name</i> transparent
<u>background-image</u>	Sets an image as the background	url(<i>URL</i>) none
<u>background-position</u>	Sets the starting position of a background image	top left top center top right center left center center center right bottom left bottom center bottom right <i>x% y%</i> <i>xpos ypos</i>

CSS Text

- CSS Text định nghĩa cho sự xuất hiện của Text
- CSS Text cho phép bạn điều khiển sự xuất hiện của text, đặt màu, tăng giảm khoảng cách giữa các kí tự, căn lề, ...

Property	Description	Values
<u>color</u>	Sets the color of a text	<i>color</i>
<u>direction</u>	Sets the text direction	ltr rtl
<u>line-height</u>	Sets the distance between lines	<i>normal</i> <i>number</i> <i>length</i> <i>%</i>
<u>letter-spacing</u>	Increase or decrease the space between characters	<i>normal</i> <i>length</i>
<u>text-align</u>	Aligns the text in an element	left right center justify
<u>text-decoration</u>	Adds decoration to text	none underline overline line-through blink

CSS Text

<u>text-indent</u>	Indents the first line of text in an element	<i>length</i> <i>%</i>
text-shadow		<i>none</i> <i>color</i> <i>length</i>
<u>text-transform</u>	Controls the letters in an element	<i>none</i> <i>capitalize</i> <i>uppercase</i> <i>lowercase</i>
unicode-bidi		<i>normal</i> <i>embed</i> <i>bidi-override</i>
<u>white-space</u>	Sets how white space inside an element is handled	<i>normal</i> <i>pre</i> <i>nowrap</i>
<u>word-spacing</u>	Increase or decrease the space between words	<i>normal</i> <i>length</i>

CSS Font

■ CSS Font định dạng font cho text

Property	Description	Values
<u>font</u>	A shorthand property for setting all of the properties for a font in one declaration	<i>font-style</i> <i>font-variant</i> <i>font-weight</i> <i>font-size/line-height</i> <i>font-family</i> caption icon menu message-box small-caption status-bar
<u>font-family</u>	A prioritized list of font family names and/or generic family names for an element	<i>family-name</i> <i>generic-family</i>
<u>font-size</u>	Sets the size of a font	xx-small x-small small medium large x-large xx-large smaller larger <i>length</i> %

CSS Font

<u>font-size-adjust</u>	Specifies an aspect value for an element that will preserve the x-height of the first-choice font	none <i>number</i>
<u>font-stretch</u>	Condenses or expands the current font-family	normal wider narrower ultra-condensed extra-condensed condensed semi-condensed semi-expanded expanded extra-expanded ultra-expanded
<u>font-style</u>	Sets the style of the font	normal italic oblique
<u>font-variant</u>	Displays text in a small-caps font or a normal font	normal small-caps

CSS Font

font-weight

Sets the weight of a font

normal
bold
bolder
lighter
100
200
300
400
500
600
700
800
900

CSS Border

- CSS Border: định nghĩa đường viền xung quanh phần tử.
- Các thuộc tính trong CSS Border cho phép định nghĩa kiểu và màu của đường viền bao quanh phần tử.

Property	Description	Values
<u>border</u>	A shorthand property for setting all of the properties for the four borders in one declaration	<i>border-width</i> <i>border-style</i> <i>border-color</i>
<u>border-bottom</u>	A shorthand property for setting all of the properties for the bottom border in one declaration	<i>border-bottom-width</i> <i>border-style</i> <i>border-color</i>
<u>border-bottom-color</u>	Sets the color of the bottom border	<i>border-color</i>
<u>border-bottom-style</u>	Sets the style of the bottom border	<i>border-style</i>
<u>border-bottom-width</u>	Sets the width of the bottom border	<i>thin</i> <i>medium</i> <i>thick</i> <i>length</i>

CSS Border

<u>border-color</u>	Sets the color of the four borders, can have from one to four colors	<i>color</i>
<u>border-left</u>	A shorthand property for setting all of the properties for the left border in one declaration	<i>border-left-width</i> <i>border-style</i> <i>border-color</i>
<u>border-left-color</u>	Sets the color of the left border	<i>border-color</i>
<u>border-left-style</u>	Sets the style of the left border	<i>border-style</i>
<u>border-left-width</u>	Sets the width of the left border	<i>thin</i> <i>medium</i> <i>thick</i> <i>length</i>
<u>border-right</u>	A shorthand property for setting all of the properties for the right border in one declaration	<i>border-right-width</i> <i>border-style</i> <i>border-color</i>
<u>border-right-color</u>	Sets the color of the right border	<i>border-color</i>

CSS Border

<u>border-right-style</u>	Sets the style of the right border	<i>border-style</i>
<u>border-right-width</u>	Sets the width of the right border	thin medium thick <i>length</i>
<u>border-style</u>	Sets the style of the four borders, can have from one to four styles	none hidden dotted dashed solid double groove ridge inset outset
<u>border-top</u>	A shorthand property for setting all of the properties for the top border in one declaration	<i>border-top-width</i> <i>border-style</i> <i>border-color</i>
<u>border-top-color</u>	Sets the color of the top border	<i>border-color</i>

CSS Border

<u>border-top-style</u>	Sets the style of the top border	<i>border-style</i>
<u>border-top-width</u>	Sets the width of the top border	thin medium thick <i>length</i>
<u>border-width</u>	A shorthand property for setting the width of the four borders in one declaration, can have from one to four values	thin medium thick <i>length</i>

CSS Margin

- CSS Margin định nghĩa khoảng cách (lề) xung quanh phần tử

Property	Description	Values
<u>margin</u>	A shorthand property for setting the margin properties in one declaration	<i>margin-top</i> <i>margin-right</i> <i>margin-bottom</i> <i>margin-left</i>
<u>margin-bottom</u>	Sets the bottom margin of an element	<i>auto</i> <i>length</i> <i>%</i>
<u>margin-left</u>	Sets the left margin of an element	<i>auto</i> <i>length</i> <i>%</i>
<u>margin-right</u>	Sets the right margin of an element	<i>auto</i> <i>length</i> <i>%</i>
<u>margin-top</u>	Sets the top margin of an element	<i>auto</i> <i>length</i> <i>%</i>

CSS Margin

Property	Description	Values
<u>margin</u>	A shorthand property for setting the margin properties in one declaration	<i>margin-top</i> <i>margin-right</i> <i>margin-bottom</i> <i>margin-left</i>
<u>margin-bottom</u>	Sets the bottom margin of an element	auto <i>length</i> %
<u>margin-left</u>	Sets the left margin of an element	auto <i>length</i> %
<u>margin-right</u>	Sets the right margin of an element	auto <i>length</i> %
<u>margin-top</u>	Sets the top margin of an element	auto <i>length</i> %

CSS Padding

- CSS Padding: định nghĩa khoảng cách giữa border và nội dung.

Property	Description	Values	IE	F	N	W3C
<u>padding</u>	A shorthand property for setting all of the padding properties in one declaration	<i>padding-top</i> <i>padding-right</i> <i>padding-bottom</i> <i>padding-left</i>	4	1	4	1
<u>padding-bottom</u>	Sets the bottom padding of an element	<i>length</i> %	4	1	4	1
<u>padding-left</u>	Sets the left padding of an element	<i>length</i> %	4	1	4	1
<u>padding-right</u>	Sets the right padding of an element	<i>length</i> %	4	1	4	1
<u>padding-top</u>	Sets the top padding of an element	<i>length</i> %	4	1	4	1

CSS List

- CSS List: cho phép đặt một danh sách, thay một danh sách khác, đặt hình cho danh sách

Property	Description	Values
<u>list-style</u>	A shorthand property for setting all of the properties for a list in one declaration	<i>list-style-type</i> <i>list-style-position</i> <i>list-style-image</i>
<u>list-style-image</u>	Sets an image as the list-item marker	<i>none</i> <i>url</i>
<u>list-style-position</u>	Sets where the list-item marker is placed in the list	<i>inside</i> <i>outside</i>

CSS List

<u>list-style-type</u>	Sets the type of the list-item marker	none disc circle square decimal decimal-leading-zero lower-roman upper-roman lower-alpha upper-alpha lower-greek lower-latin upper-latin hebrew armenian georgian cjk-ideographic hiragana katakana hiragana-iroha katakana-iroha
marker-offset		auto <i>length</i>

Định dạng tài liệu XML dùng bảng định kiểu (CSS)

Ví dụ

- Ta có một file data.xml như sau:

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

```
<document>
```

```
    <title>Giáo trình XML</title>
```

```
    <author>Nguyễn Thiện Bằng</author>
```

```
    <paragraph>
```

Chào mừng các bạn đến với XML, XML là một ngôn ngữ dùng để quản lý dữ liệu ở hình thức ngắn gọn, dễ dàng cho việc quản lý.

```
    </paragraph>
```

```
    <paragraph>
```

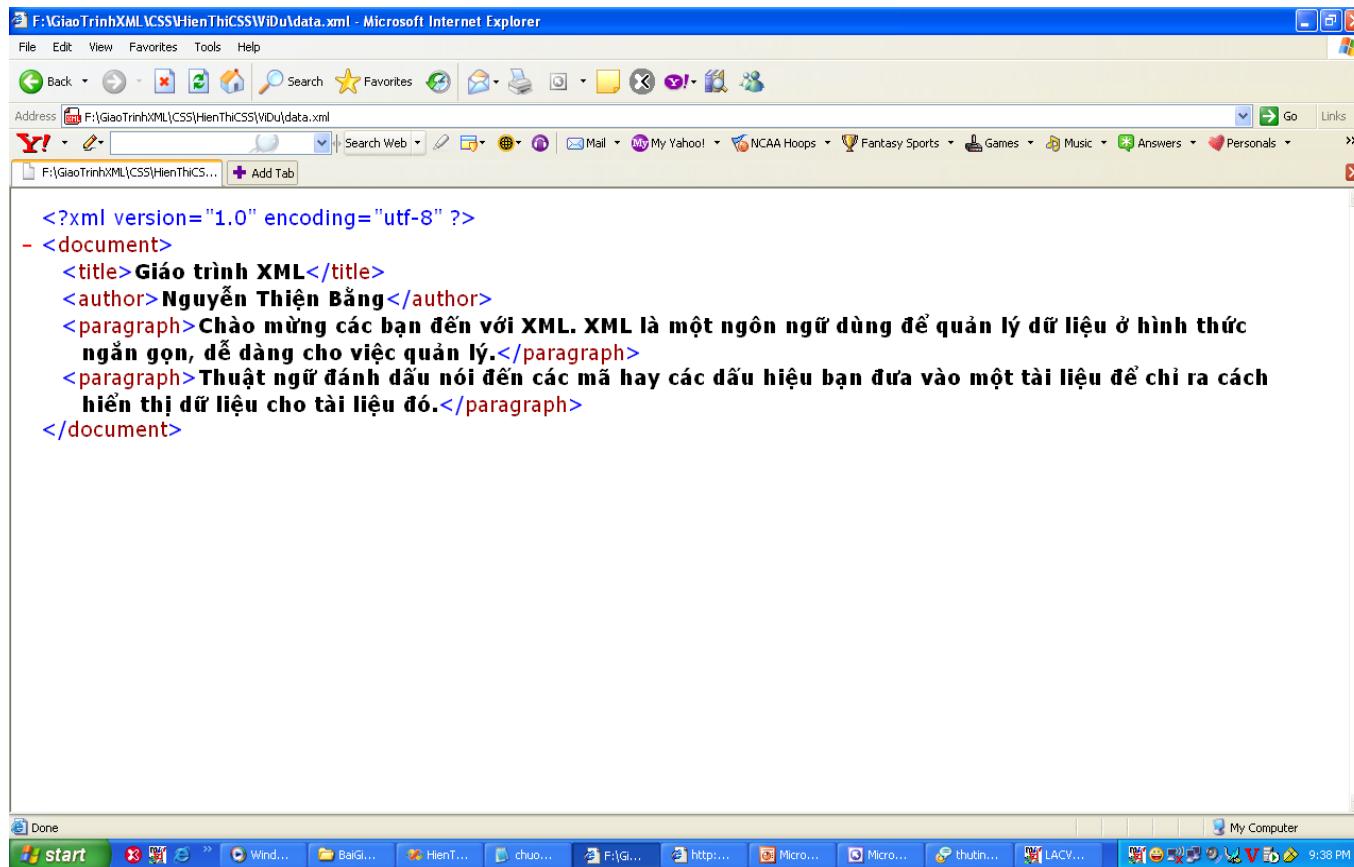
Thuật ngữ đánh dấu nói đến các mã hay các dấu hiệu bạn đưa vào một tài liệu để chỉ ra cách hiển thị dữ liệu cho tài liệu đó.

```
    </paragraph>
```

```
</document>
```

Ví dụ

■ Dùng trình duyệt hiển thị tài liệu data.xml:



Giới thiệu CSS

- CSS được chuẩn hóa bởi W3C, có 3 cấp CSS có sẵn: CSS1, CSS2, CSS3. CSS1 có tất cả những gì cần cho hiển thị tài liệu XML.
- Các trình duyệt IE hỗ trợ rất tốt cho CSS1.
- CSS chủ yếu dùng bảng định kiểu. Bảng định kiểu là tập hợp các quy tắc về kiểu chỉ ra các định dạng phần tử.

Giới thiệu

- Ví dụ: Để chỉ ra quy tắc cho phần tử <title> là chữ đậm, cỡ chữ 36, canh giữa ta định nghĩa CSS như sau:

```
title {display:block; font-size: 36pt; font-weight: bold; text-align: center;}
```

- Trong đó: title được gọi là bộ chọn. Display, font-size, font-weight, text-align là thuộc tính CSS

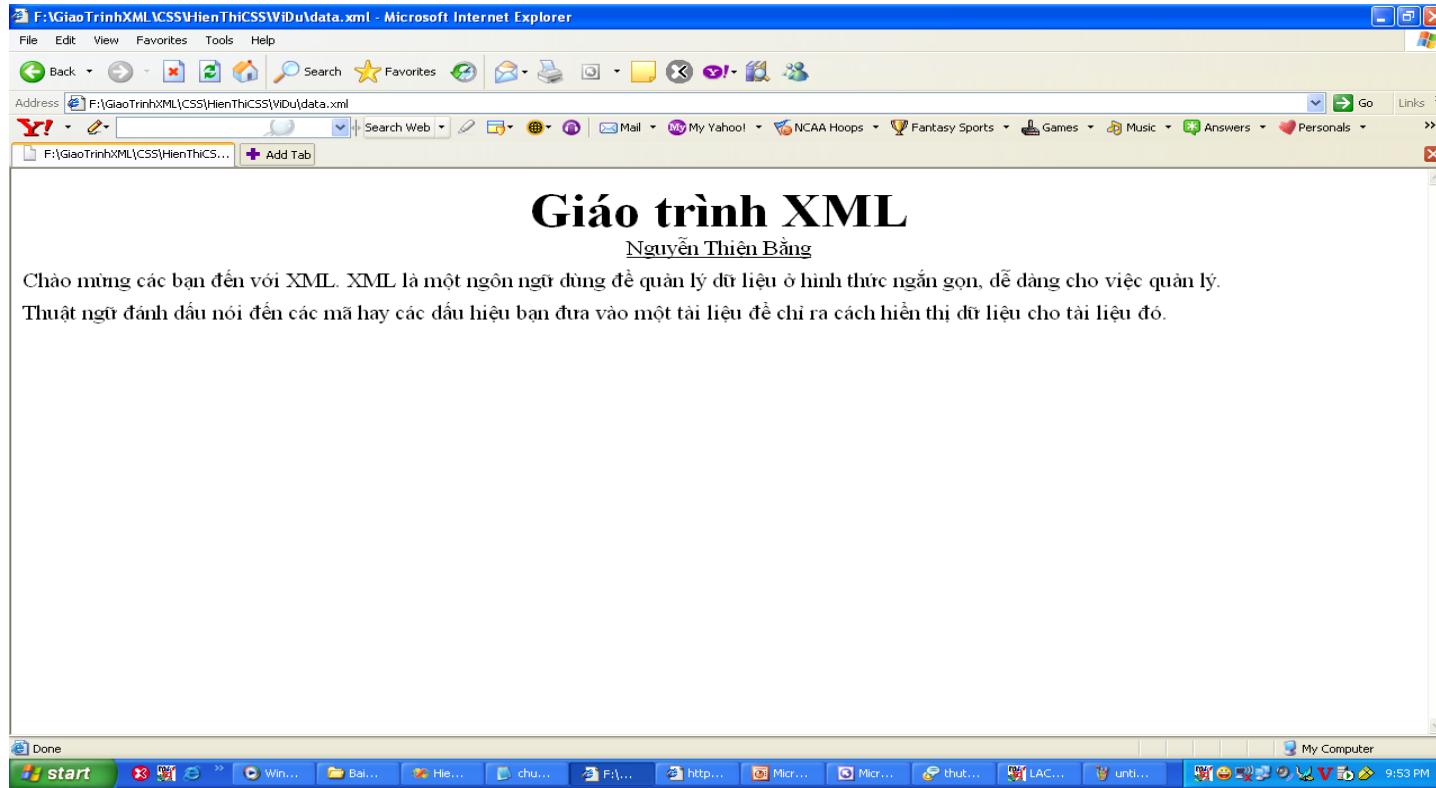
Giới thiệu

- Tạo bảng định kiểu cho ví dụ trên như sau:

```
title
{
    display:block;
    font-size: 36pt;
    font-weight: bold;
    text-align:center;
}
author
{
    display:block;
    font-size:16pt;
    text-align:center;
    text-decoration:underline
}
paragraph
{
    display:block;
    margin-top:10
}
```

Giới thiệu

- Hiển thị trên trình duyệt ví dụ trên khi kết hợp với bảng định kiểu CSS:



Kết hợp CSS với tài liệu XML

- Có hai cách để đưa CSS vào tài liệu HTML: dùng bảng định kiểu chung với tài liệu HTML hoặc dùng bảng định kiểu lưu thành một file riêng biệt với HTML.
- Với XML thì chỉ có một cách đưa CSS vào tài liệu là lưu thành tập tin riêng và đưa vào tài liệu XML chỉ thị xử lý như sau:

```
<?xmlstylesheet type="text/css" href=URI?>
```

Ví dụ trước, chỉ thị xử lý trong tài liệu XML là:

```
<?xmlstylesheet type="text/css" href="data.css"?>
```

Tạo bộ chọn cho bảng định kiểu

■ Quy tắc tạo bộ:

- ❑ Để chỉ ra phần tử nào muốn định dạng bạn dùng bộ chọn trong bảng định kiểu CSS. Bộ chọn có tên là tên phần tử bạn muốn định kiểu.

Ví dụ: đặc tả title như sau:

```
title { display:block; font-size: 36pt; font-weight: bold; text-align:center;}
```

- ❑ Nếu các phần tử có cùng đặc tả giống nhau, bạn có thể nhóm chúng lại và phân cách bằng dấu “,”.

Ví dụ: giả sử title và book có cùng đặc tả

```
title, book { display:block; font-size: 36pt; font-weight: bold; text-align:center;}
```

Tạo bộ chọn cho bảng định kiểu

■ Tạo lớp định kiểu:

- ❑ Có thể không cần chỉ ra tên phần tử hoặc các phần tử theo thứ tự để tạo bộ chọn. Thay vào đó bạn dùng định nghĩa lớp định kiểu. Tạo lớp bằng cách có dấu “.” trước tên lớp.

Ví dụ: trong file data.css, ta định nghĩa lớp standout như sau:

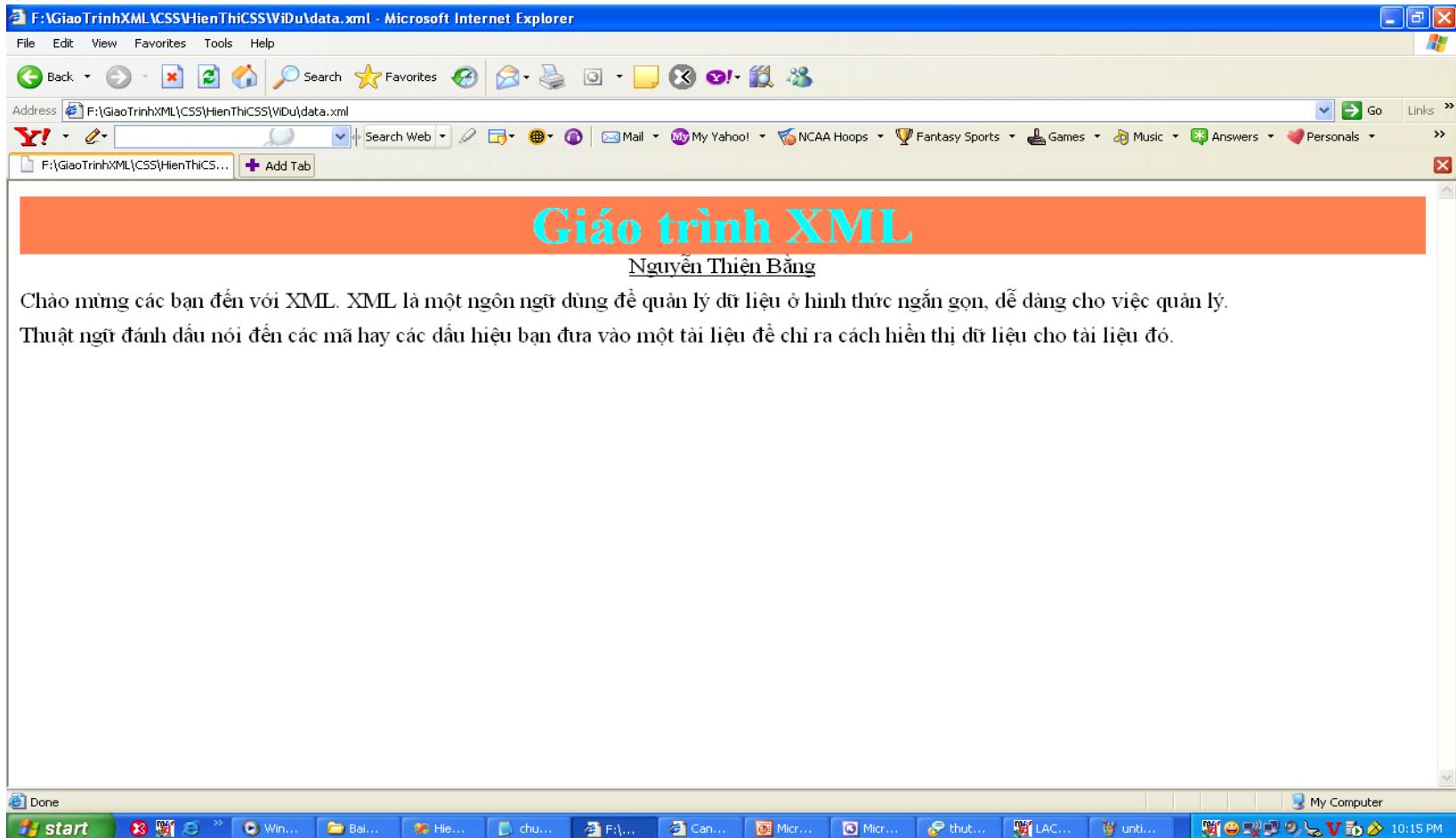
```
.standout { color:cyan; background-color: coral}
```

Áp dụng lớp vào phần tử title của tài liệu data.xml như sau:

```
<title class="standout"> Giáo trình XML</title>
```

Kết quả khi duyệt:

Tạo bộ chọn cho bảng định kiểu



Tạo bộ chọn cho bảng định kiểu

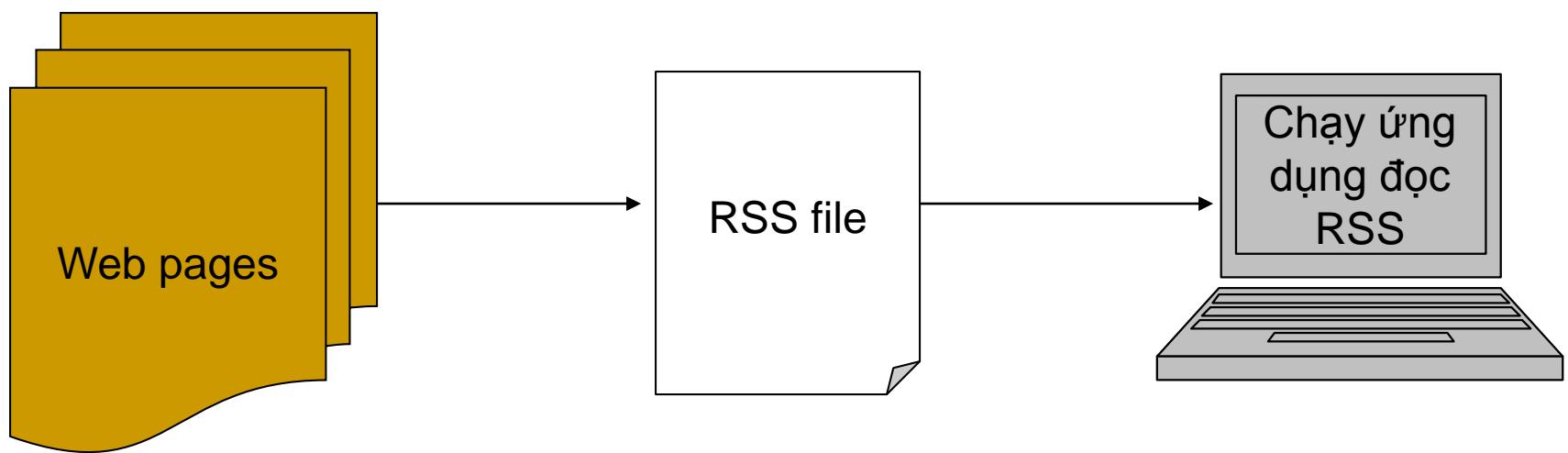
- Định dạng bằng ID: ngoài cách tạo lớp định kiểu ta còn có thể chọn phần tử cần định dạng bằng định danh ID và sử dụng giá trị ID đó cho phần tử.
 - ❑ Cú pháp: elementName#idValue
 - ❑ Ví dụ: trong ví dụ trên, trong file data.css ta định nghĩa paragraph như sau:
paragraph#first{text-indent:40; margin-top: 30}
trong tài liệu data.xml ta sửa như sau:
<paragraph id="first"> ..</paragraph>

Ứng dụng đọc tin nhanh sử dụng RSS

Giới thiệu

- RSS là gì?
 - RSS là một định dạng tập tin thuộc họ XML (ngôn ngữ đánh dấu mở rộng, một chuẩn dùng để mô tả dữ liệu)
- Được dùng trong việc chia sẻ tin tức Web (Web syndication) được dùng bởi nhiều website tin tức và nhật ký trực tuyến

Mô hình hoạt động



Ví dụ

CNN RSS

CNN.com

Subscribe to CNN's RSS (Really Simple Syndication) feeds to get news delivered directly to your desktop!

To view one of the CNN feeds in your RSS Aggregator ([About RSS Aggregators](#)):

1. Copy the URL/shortcut that corresponds to the topic that interests you.
2. Paste the URL into your reader.

My Yahoo! users:

1. Click on the "Add to My Yahoo!" button.
2. Follow the instructions for adding the feed to your My Yahoo! page.

NEW: CNN.com now offers [podcasting feeds](#). 

Please note that by accessing CNN RSS feeds, you agree to our [terms of use](#).

[What is RSS? | How do I access RSS?](#)

Title	Copy URLs to RSS Reader	
Top Stories	http://rss.cnn.com/rss/edition.rss	 
World	http://rss.cnn.com/rss/edition_world.rss	 
Africa	http://rss.cnn.com/rss/edition_africa.rss	 
Americas	http://rss.cnn.com/rss/edition_americas.rss	 
Asia	http://rss.cnn.com/rss/edition_asia.rss	 
Europe	http://rss.cnn.com/rss/edition_europe.rss	 
Middle East	http://rss.cnn.com/rss/edition_meast.rss	 
U.S.	http://rss.cnn.com/rss/edition_us.rss	 
World Business	http://rss.cnn.com/rss/edition_business.rss	 

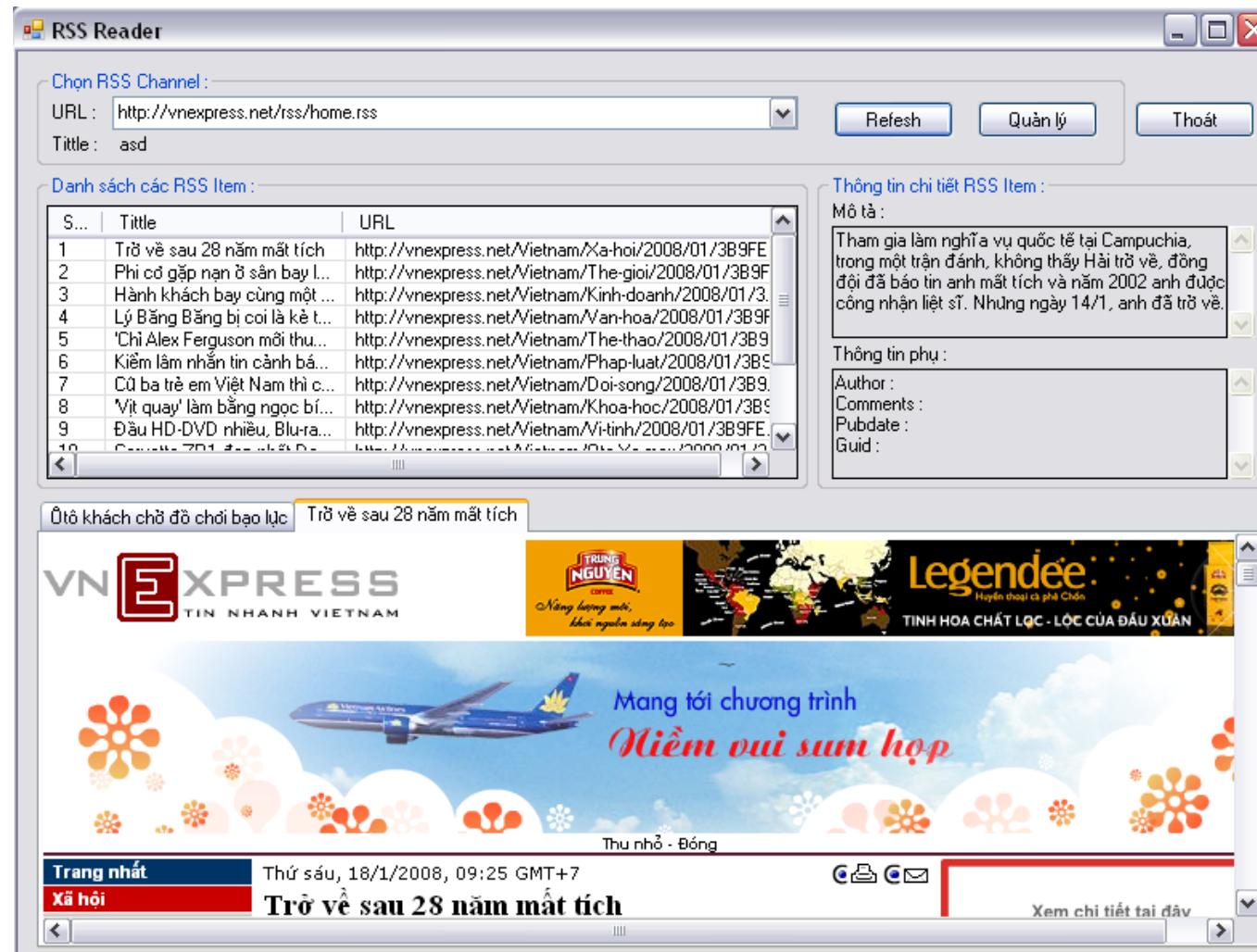
Ví dụ

■ Vnexpress.net

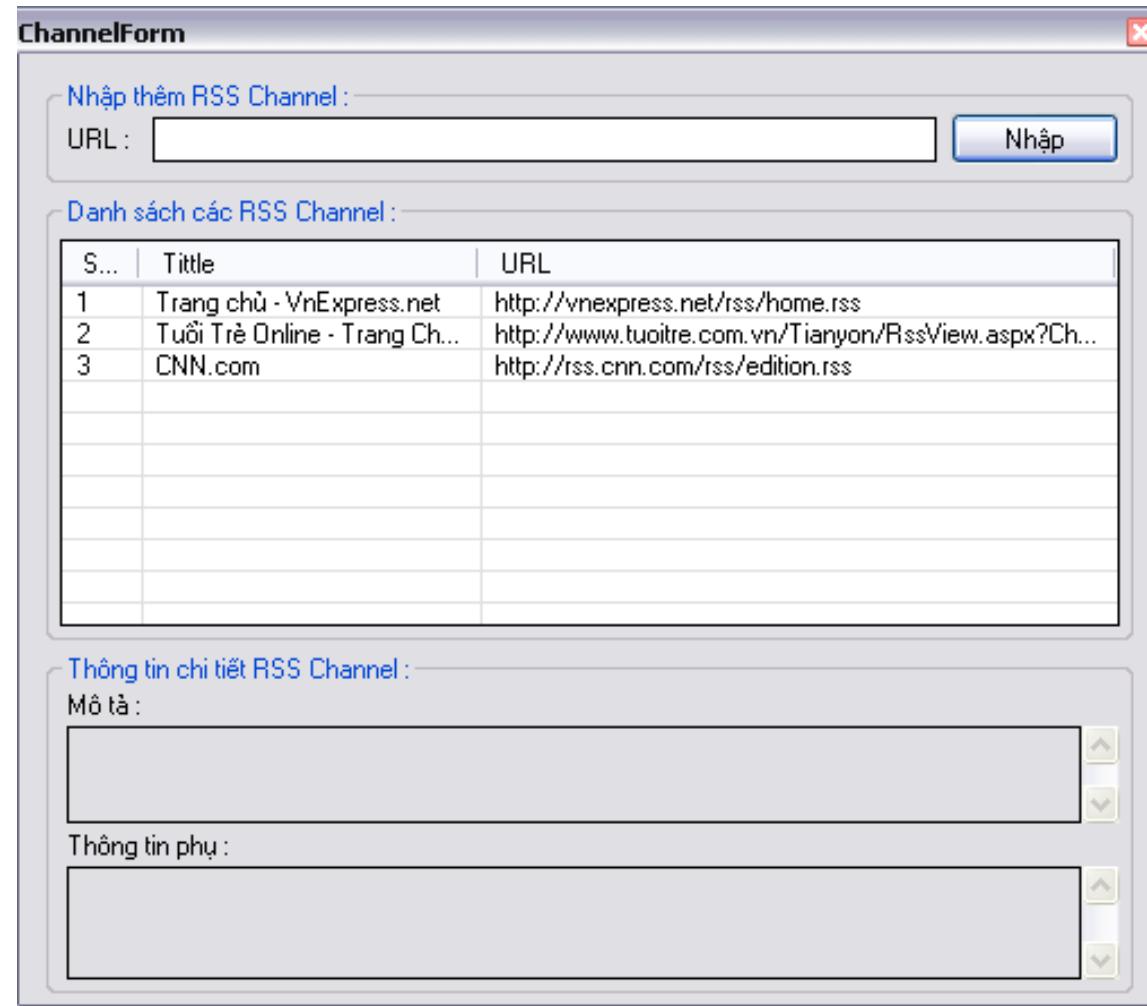
Các kênh RSS do VnExpress cung cấp

		Trang chủ	(http://vnexpress.net/rss/home.rss)
		Xã hội	(http://vnexpress.net/rss/xa-hoi.rss)
		Thế giới	(http://vnexpress.net/rss/the-gioi.rss)
		Kinh doanh	(http://vnexpress.net/rss/kinh-doanh.rss)
		Văn hóa	(http://vnexpress.net/rss/van-hoa.rss)
		Thể thao	(http://vnexpress.net/rss/the-thao.rss)
		Pháp luật	(http://vnexpress.net/rss/phap-luat.rss)
		Đời sống	(http://vnexpress.net/rss/doi-song.rss)
		Khoa học	(http://vnexpress.net/rss/khoa-hoc.rss)
		Vì tinh	(http://vnexpress.net/rss/vi-tinh.rss)
		Ôtô - Xe máy	(http://vnexpress.net/rss/oto-xe-may.rss)
		Bạn đọc viết	(http://vnexpress.net/rss/ban-doc-viet.rss)
		Tâm sự	(http://vnexpress.net/rss/ban-doc-viet-tam-su.rss)
		Cười	(http://vnexpress.net/rss/cuoi.rss)

Chương trình minh họa



Chương trình minh họa



Cấu trúc file RSS

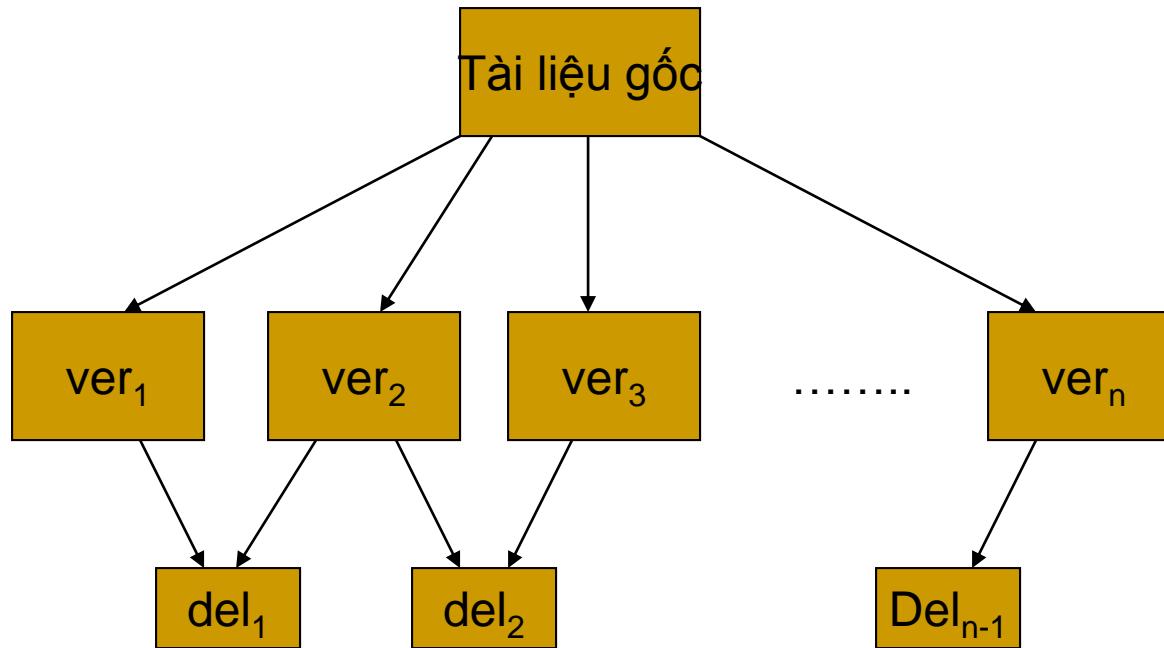
```
<?xml version="1.0" ?>
<rss version="0.91">
  <channel>
    <title>RSS Tutorial </title>
    <link>http://www.linktothestory.com</link>
    <description> The RSS Tutorial</description>

    <item>
      <title>RSS Syndication </title>
      <link>http://www.linktothestory.com</link>
      <description>Blah,Blah...</description>
    </item>

    <item>
      <title>Technology in Crisis??</title>
      <link>http://www.linktothestory.com</link>
      <description>The problem with technology in Schools</description>
    </item>

  </channel>
</rss>
```

Version Control System

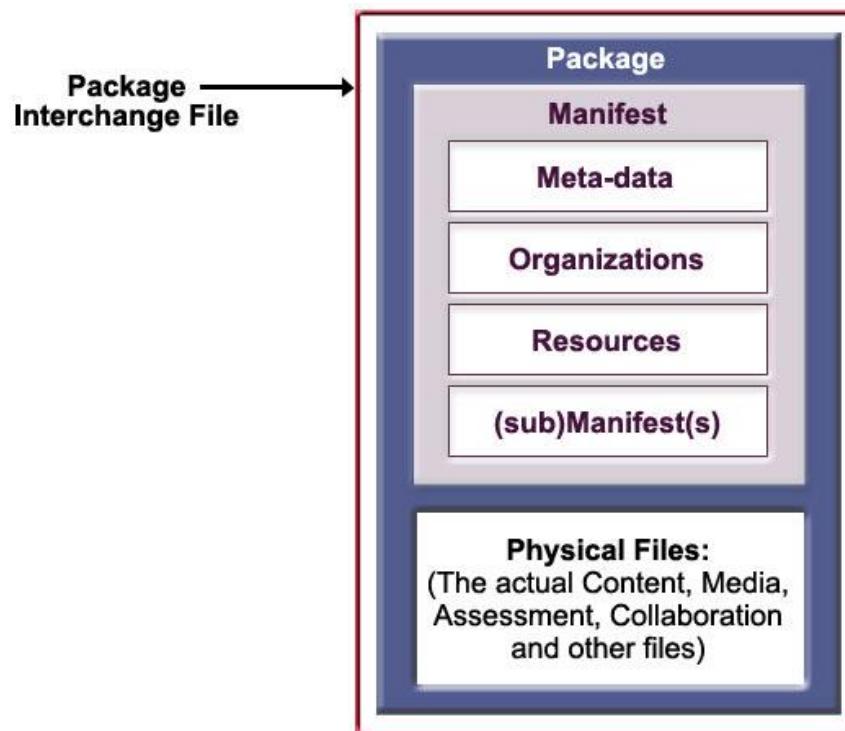


Version Control System

- Áp dụng vào các hệ thống đa người dùng:
 - Các ứng dụng quản trị nội dung.
 - Các ứng dụng quản lý tài liệu điện tử.
- Các ứng dụng cần có sự thay đổi cấu trúc thường xuyên.

SCORM

- Tập các chuẩn và đặc tả được xây dựng nhằm cung cấp các quy tắc chuẩn hóa cho việc triển khai việc dạy và học tập qua mạng.



Các thành phần của Manifest :

Meta-data : Siêu dữ liệu mô tả về gói nội dung

Organization : mô tả cấu trúc nội dung hoặc tổ chức các tài nguyên học tập

Resources: định nghĩa các tài nguyên học tập được gộp vào trong gói nội dung.

Visualization

- Visualization là một kĩ thuật biểu diễn thông tin một cách trực quan sinh động. Dữ liệu được biểu diễn dạng XML.

Công nghệ XML và WEB ngữ nghĩa

Introduction to The Semantic WEB

Thông tin và internet

- World Wide Web (WWW) là môi trường tốt cho việc biểu diễn và truy cập thông tin dạng số.
- Thông tin trên WWW được biểu diễn chủ yếu dưới dạng ngôn ngữ tự nhiên (các trang Web trên ngôn ngữ HTML).
- Máy tính và người hiểu khác nhau!
- **Mong muốn: machine-readable & machine analysis**

Bài toán tìm kiếm thông tin

Information Retrieval

- Information retrieval (IR) là quá trình tìm kiếm dữ liệu (thường là tìm dưới dạng một đoạn **văn bản**) từ một **tập hợp lớn** các đối tượng không có cấu trúc tường minh (thường là text **lưu trữ trong máy tính**) nhằm đáp ứng một nhu cầu về thông tin



Công cụ tìm kiếm trực tuyến

The screenshot shows a web browser window with the following details:

- Address Bar:** http://www.google.com.vn/#scion=psy...
iwi-iuk.org
seantic web là gi... x
- Menu Bar:** File, Edit, View, Favorites, Tools, Help
- Toolbar:** +Bạn, Tim kiem, Hình ảnh, Video, Tin tức, Dịch, Gmail, Khác
- Right Sidebar:** Đăng nhập, Settings icon
- Search Query:** seantic web là gì?
- Search Results Summary:** Khoảng 97.500 kết quả (0,12 giây)
- Info Box (Yellow):** Chúng tôi sắp thay đổi chính sách bảo mật và các điều khoản. Thay đổi có ý nghĩa quan trọng.
[Tìm hiểu thêm](#) | [Loại bỏ](#)
- Left Sidebar (Mọi thứ):**
 - Hình ảnh
 - Video
 - Tin tức
 - Thảo luận
 - Nhiều hơn
- Result Preview:** Hiển thị kết quả cho [semantic web là gì?](#)
Tim kiem thay thế cho seantic web là gi?
- First Result:** [Tổng quan về Semantic Web](#)
www.phpvn.org/index.php?topic=119.0;wap2
9 Tháng Bảy 2008 – Mục tiêu ban đầu của Semantic Web là để hỗ trợ người dùng tìm kiếm thông tin ... kiem sẽ không thể nào biết được thông tin trên file HTML đó biểu diễn cái gì.
- Second Result:** [Semantic Web - Huyền thoại về thế hệ web trong tương lai](#) - 10 Tháng Bảy 2008
[Protege](#) - 10 Tháng Bảy 2008
[XML, XML NS và XML Schema](#) - 10 Tháng Bảy 2008
[Các kết quả khác từ phpvn.org »](#)
- Bottom Left:** Hà Nội, Thay đổi vị trí
- Bottom Right:** Semantic Web Intro
www.slideshare.net/nttuyen/semantic-web-intro - Hoa Kỳ

Thực chất google làm gì?

- Gửi đi một yêu cầu (*query*) như sau
- http://www.google.com.vn/#sclient=psy-ab&hl=vi&source=hp&q=seantic+web+l%C3%A00+gi%3F&pbx=1&oq=seantic+web+l%C3%A00+gi%3F&aq=f&aqi=&aql=&gs_sm=e&gs_upl=910l13129l0l14005l24l20l3l0l0l1l827l4892l0.13.3.2.0.1.1l23l0&bav=on.2,or.r_gc.r_pw.,cf.osb&fp=2b8791cc67af876b&biw=1280&bih=683

Hiện nay làm thế nào để máy tính hiểu yêu cầu?

- Boolean retrieval: thiết kế và cấu trúc dữ liệu cho một hệ thống thu thập thông tin đơn giản

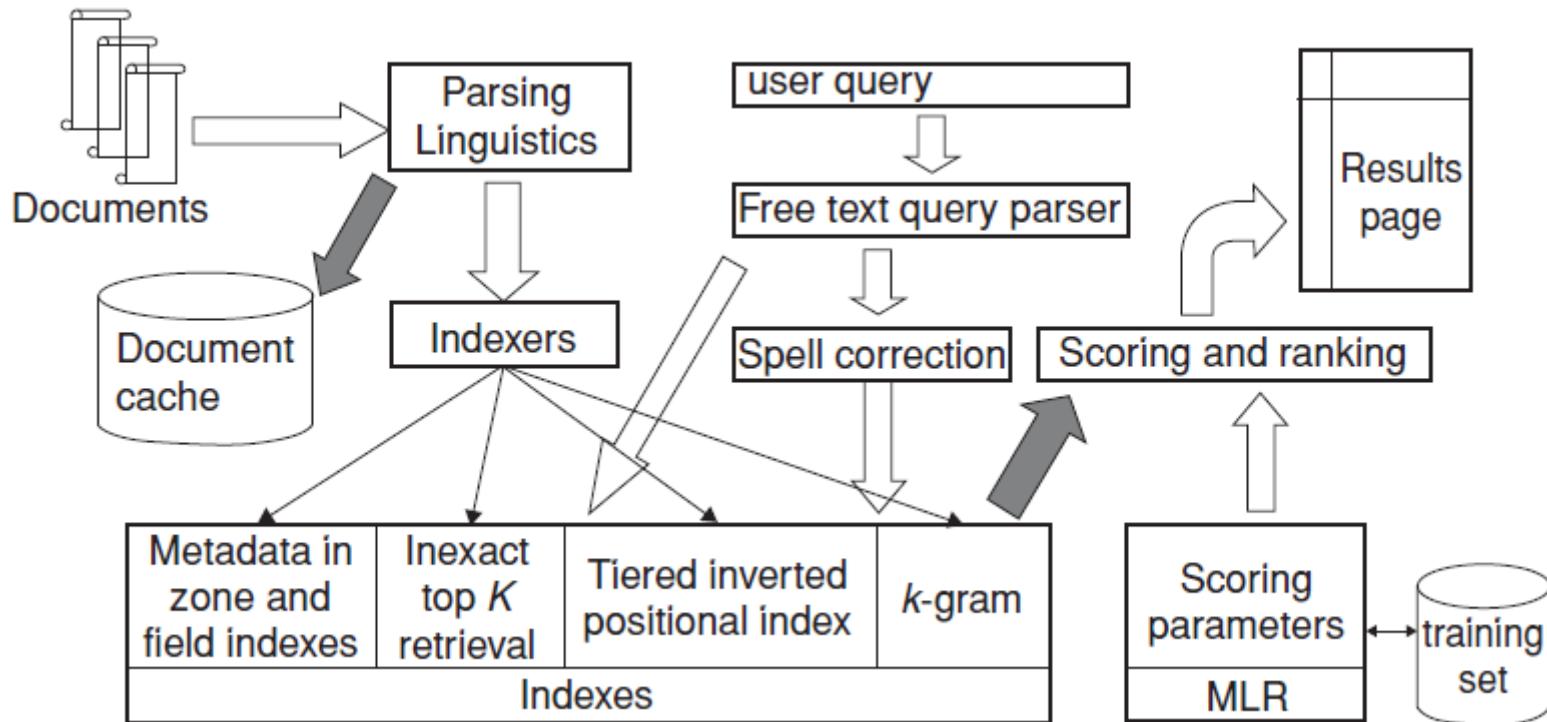
Quy trình Hoạt động:

1. Xây dựng Boolean model, mô hình logic dùng để thu thập thông tin.
2. Biểu diễn yêu cầu dưới dạng biểu thức logic
3. Search engine trả về tất cả các tài liệu thỏa mãn biểu thức logic

Kho dữ liệu khổng lồ

- Với 1M tài liệu văn bản lưu trữ, mỗi văn bản có khoảng 1000 từ tách rời
- Suy ra: tổng số 1 tỷ từ tách rời (tính cả trùng nhau)
- Trung bình k|h|o|ả|n|g| 6 bytes cho một từ tách rời ra (tính cả dấu cách dấu biểu cảm)
- Suy ra kho dữ liệu khoảng 6GB
- Giả sử chỉ có khoảng 500K từ -> cần lập bảng thống kê với số lượng:
 $500000 * 1000000 = 5 * 10^{11}$

Search engine



Vấn đề cấu trúc dữ liệu text

- **HTML** (Hyper text markup language)
- **XML** (Extensible Markup language)
- **SIML** (Synchronized Multimedia Integration Language) <http://www.w3.org/Audio>

HTML

- HTML là chữ viết tắt của Hyper Text Markup Language (Ngôn ngữ hiển thị siêu văn bản).
- - Một file HTML là một file text bao gồm những tag nhỏ
- - Những tag hiển thị nói cho trình duyệt biết nó phải hiển thị trang đó như thế nào
- - Một file HTML phải có phần mở rộng là .htm hoặc .html
- - Một file HTML có thể được tạo bởi một trình soạn thảo đơn giản.
- <p>Đây là đoạn văn</p>
<p>Đây là một đoạn văn khác</p>
- Vấn đề của HTML là nó được thiết kế như trong ý nghĩ của con người. Cho dù thông tin trên HTML không được thể hiện trên một trình duyệt thì chúng ta vẫn đoán biết
- Con người chúng ta đều có trí thông minh để hiểu được ý nghĩa và mục đích của hầu hết các văn bản. Tuy nhiên một cỗ máy lại không như thế. Khi các thẻ trong tài liệu này chỉ cho một trình duyệt cách thể hiện thông tin, thì bản thân các thẻ lại không chỉ cho trình duyệt thông tin đó là gì

XML

- *Lớp* XML Extensible Markup Language là một mở rộng của ngôn ngữ đánh dấu cho các cấu trúc tài liệu bất kỳ
- *I just got a new pet dog.*
- <sentence>
- <person href="http://aaronsw.com/">I</person>
- just got a new pet
- <animal>dog</animal>.
- </sentence>

Web có ngữ nghĩa

- Người sáng lập: Tim Berners_Lee giám đốc tổ chức World Wide Web Consortium (<http://www.w3c.org>)



Định nghĩa của Tim Berners – Lee

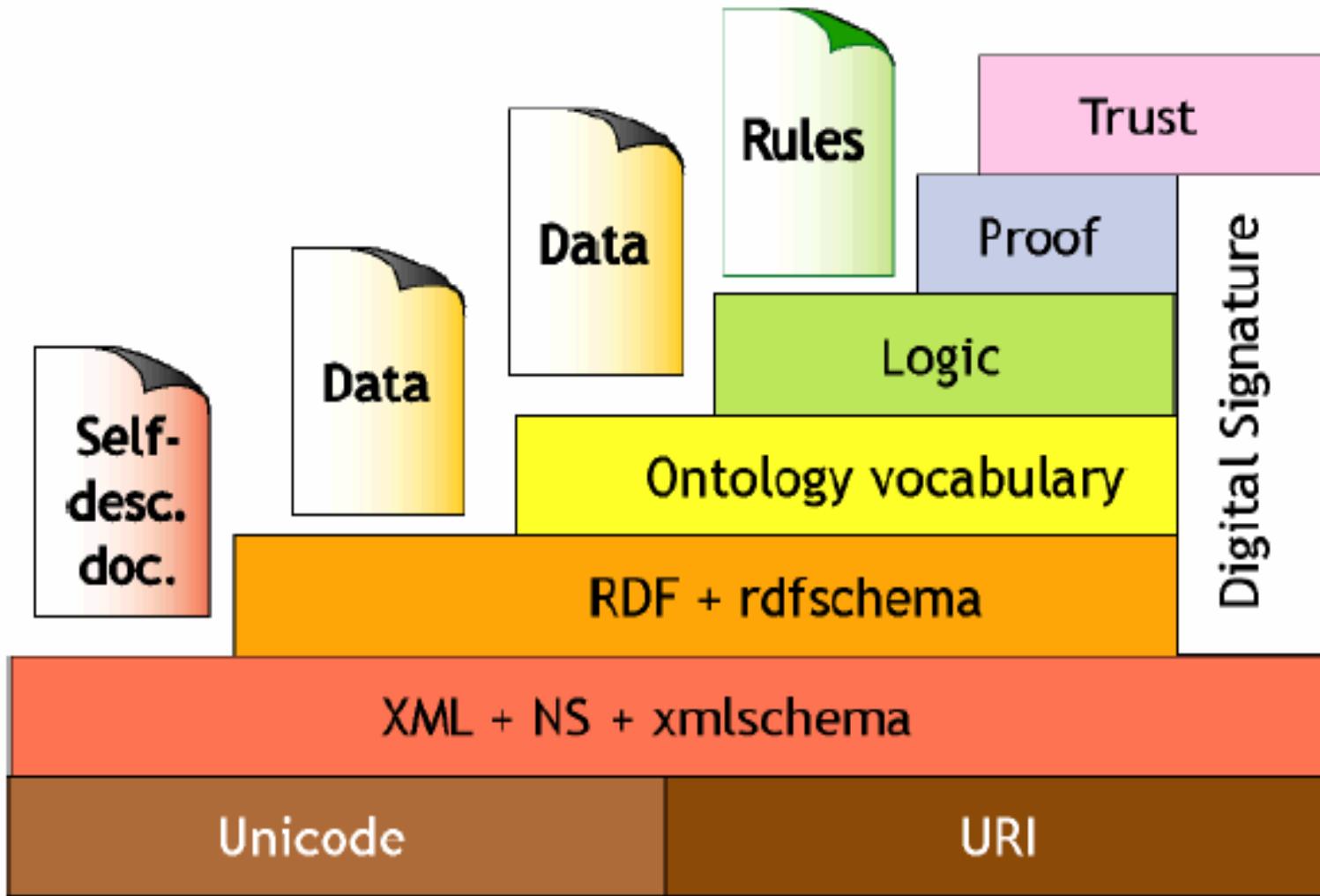
- **Semantic web** như một sự mở rộng của web hiện tại mà trong đó thông tin được xử lý một cách tự động bằng máy tính, làm cho máy tính và con người có thể hợp tác với nhau.

Semantic là gì?

- Tôi yêu em = Tui iu iem = I love you = Tôi iêu em= Anh yêu em = Em yêu anh =



CẤU TRÚC



CHI TIẾT

- **Tầng Unicode & URI:** Nhằm đảm bảo việc sử dụng tập kí tự quốc tế và cung cấp phương tiện nhằm định danh các đối tượng trong Semantic Web.
- **Tầng XML, Namespace & XMLSchema:** Tầng này bảo đảm rằng chúng ta có thể tích hợp các định nghĩa Semantic Web với các chuẩn dựa trên XML khác.
- **Tầng RDF & RDFS [RDFSchema]:** Tầng này dùng siêu dữ liệu để mô tả tài liệu trên web mà máy tính có thể hiểu được. Đây cũng là lớp mà chúng ta có thể gán các kiểu cho các tài nguyên và liên kết. Và cũng là lớp ***quan trọng nhất*** trong Semantic Web.

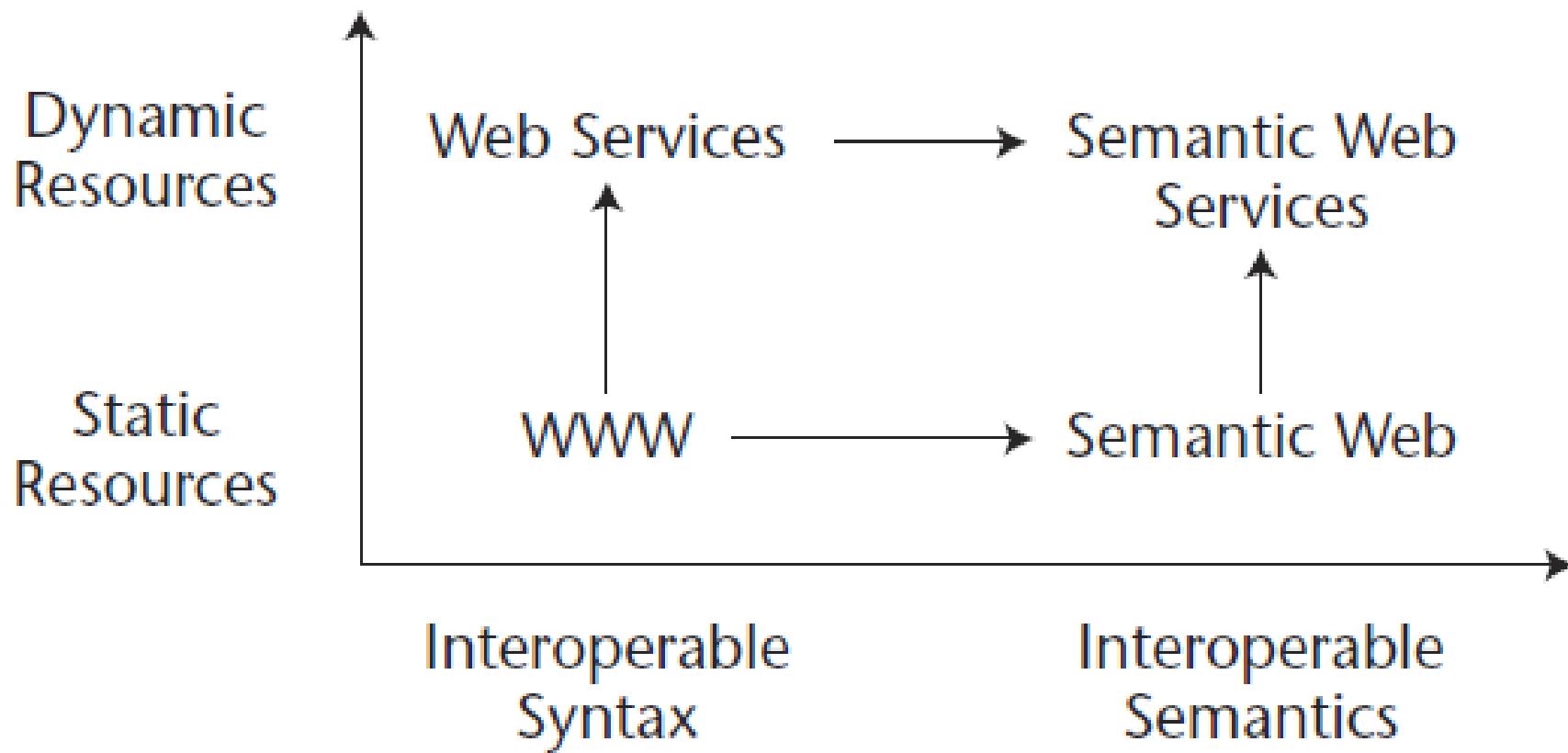
CHI TIẾT

- **Tầng Ontology** : cung cấp từ vựng chung cho việc trao đổi thông tin giữa các ứng dụng và dịch vụ Web.
- **Lớp Digital Signature**: Lớp này được dùng cho các tầng(tầng RDF –RSFS, Ontonogy, Logic, Proof) được dùng để xác định chủ thể của tài liệu, nhằm đảm bảo độ tin cậy của tài.
- **Tầng Logic**: Tầng logic được xem như là một cơ sở luật trên Semantic Web.
- **Tầng Proof**: dùng để chứng minh các suy diễn của hệ thống bằng cách liên kết các dữ kiện.
- **Tầng Trust**: Trust engine là một hệ thống đang được xây dựng dựa trên nền tảng của chữ ký điện tử.

ĐỂ CÓ WEB NGỮ NGHĨA CẦN GÌ?

- Ontology và các ngôn ngữ dùng để biểu diễn ngữ nghĩa thông tin.
- Các công cụ tạo nên phần ngữ nghĩa cũng như cấu trúc hạ tầng của Web có ngữ nghĩa.
- Các ứng dụng sử dụng Web có ngữ nghĩa.

SEMANTIC WEB SERVICES VÀ WEB SERVICES



TỰ HỌC

- Nghe bài giảng trên youtube
- <http://www.youtube.com/watch?v=rhgUDGtT2EM&feature=related>

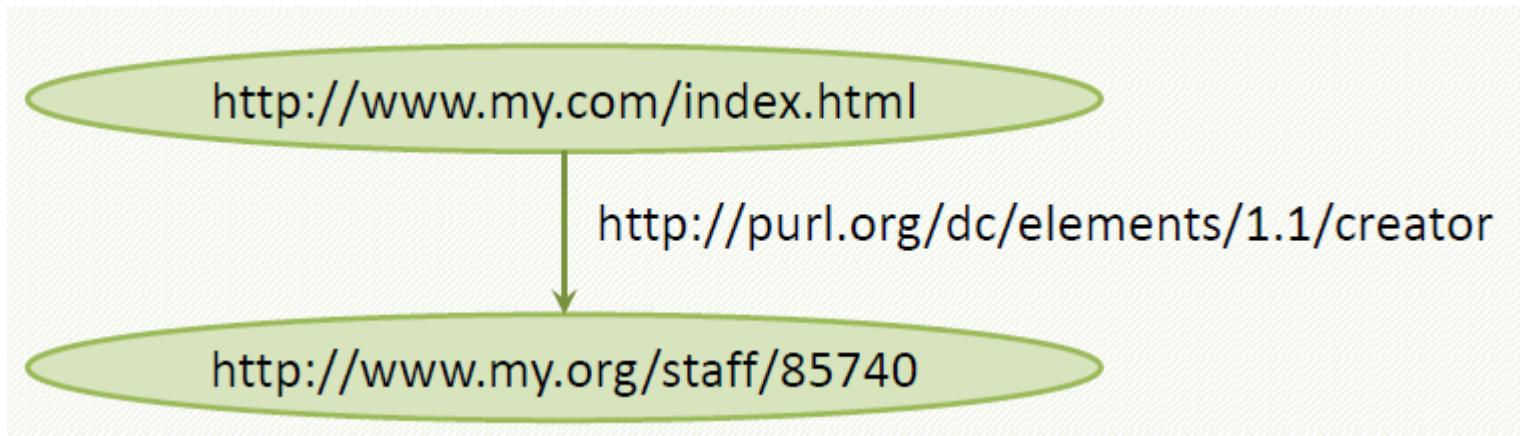
RDF Ôn tập & thực hành

RDF

- RDF = Resource Description Framework
- <http://www.w3.org/TR/rdf-primer/>
- Triples : Subject Predicate Object
- Graph model

Ví dụ

- <<http://www.my.com/index.html>>
- <<http://purl.org/dc/elements/1.1/creator>>
- <<http://www.my.org/staff/85740>>

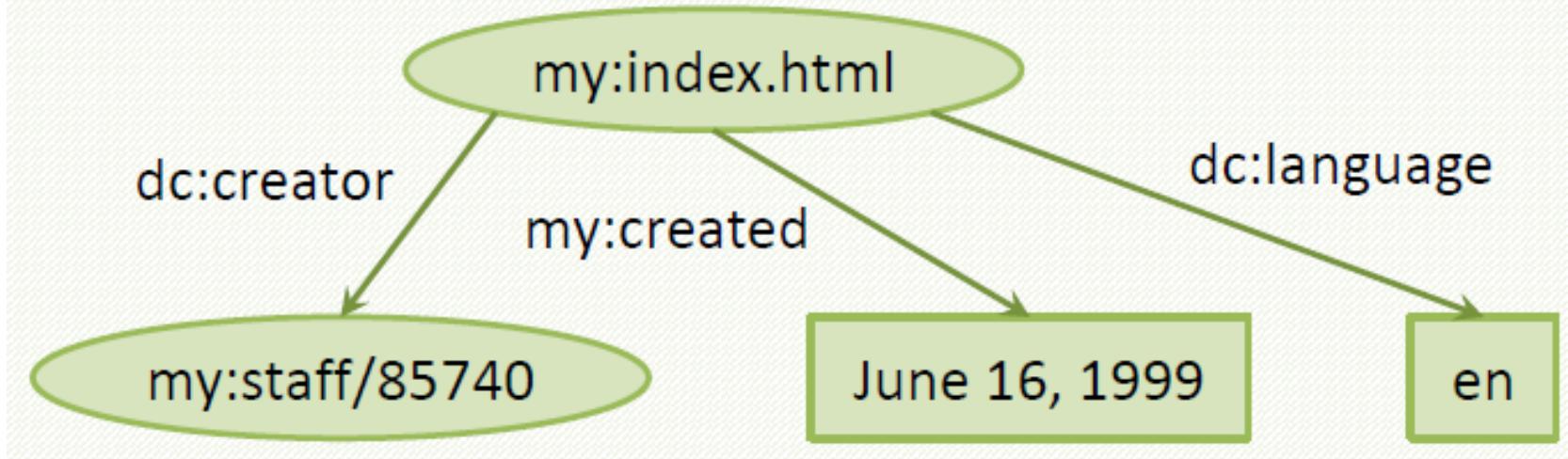


Resources

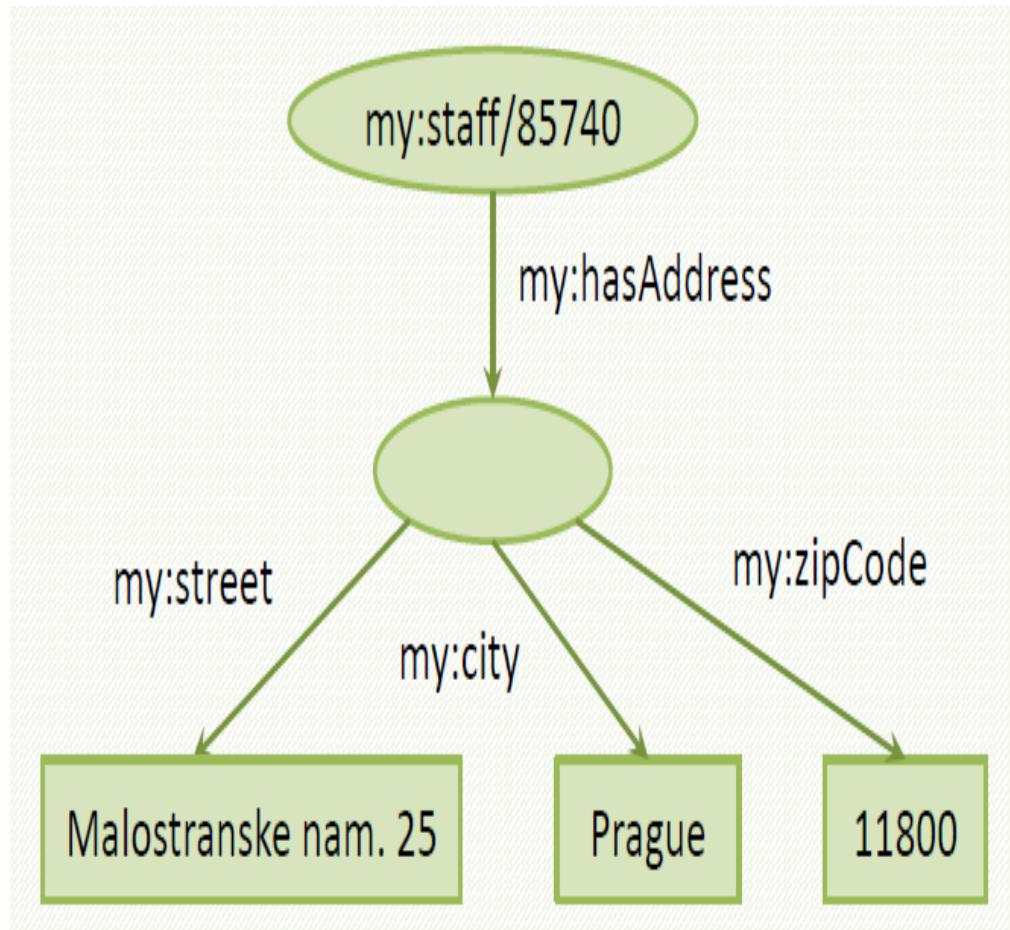
- URI
- Qualified names
 - rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
 - rdfs: <http://www.w3.org/2000/01/rdf-schema#>
 - dc: <http://purl.org/dc/elements/1.1/>
- Literals

Các ký hiệu: hình oval, chữ nhật

```
my:index.html dc:creator exstaff:85740 .  
my:index.html my:created "June 16, 1999"  
my:index.html dc:language "en" .
```



Blank nodes



exstaff:85740 my:hasAddress _:a1 .
_:a1 my:street "Malostranske nam. 25" .
_:a1 my:city "Prague" .
_:a1 my:zipCode "11800"

RDF/XML

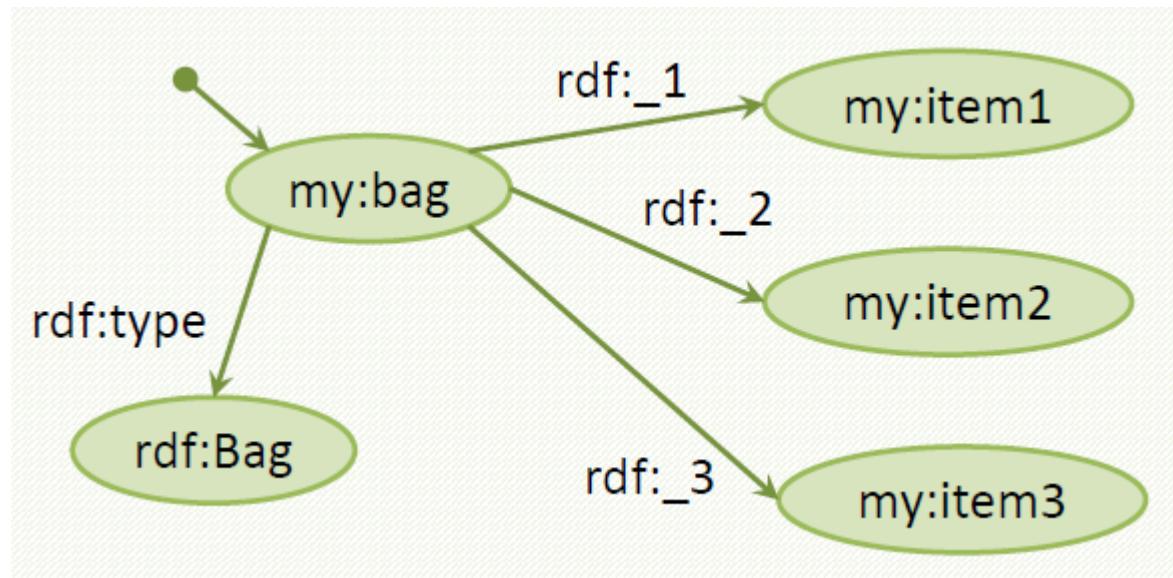
```
<rdf:RDF ...>
<rdf:Description rdf:about="SubjectResource">
<PredicateResource>ObjectLiteral</PredicateResou
rce>
<PredicateResource
  rdf:resource="ObjectResource"/>
...
</rdf:Description>
...
</rdf:RDF>
```

Biểu diễn nốt trống (blank nodes)

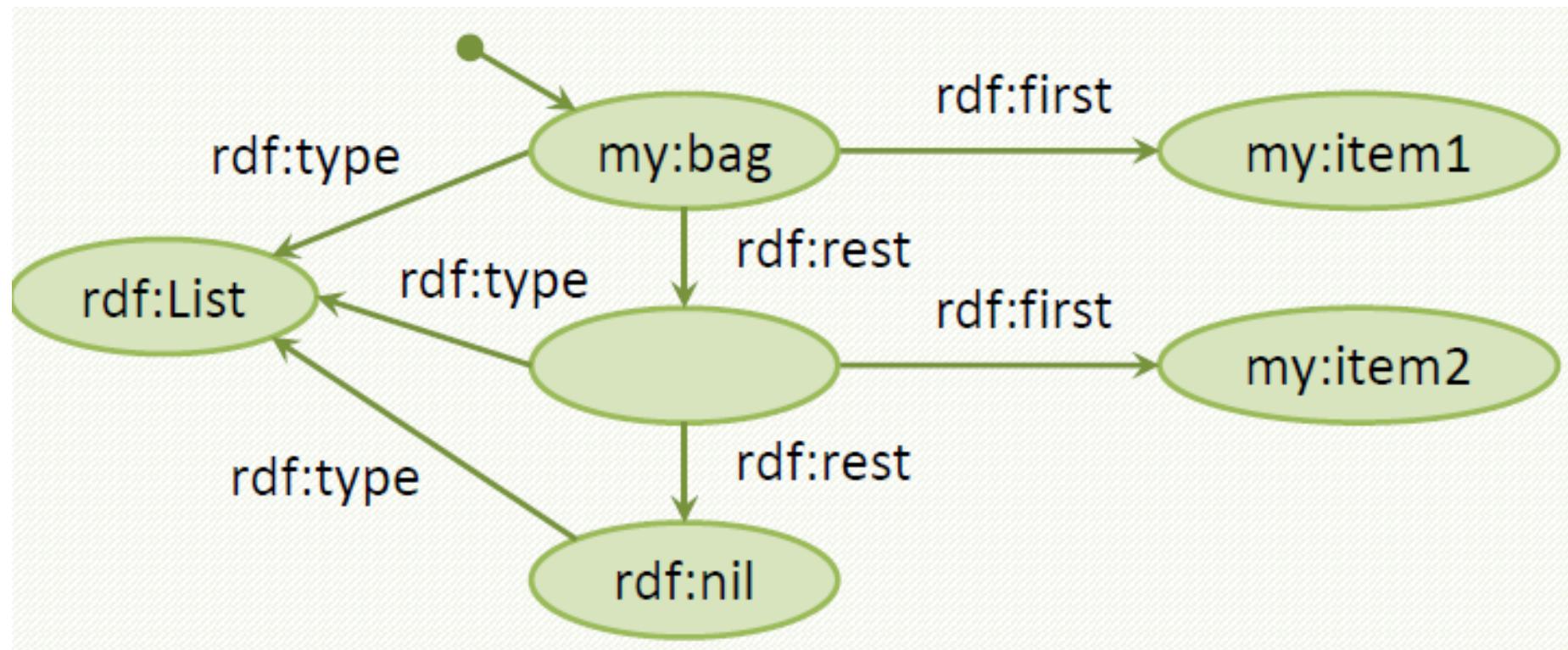
```
<rdf:RDF ...>
<rdf:Description rdf:about="SubjectResource">
<PredicateResource rdf:nodeID="BlankNode"/>
</rdf:Description>
<rdf:Description rdf:nodeID="BlankNode">
...
</rdf:Description>
...
</rdf:RDF>
```

Containers

- Dùng để biểu diễn nhóm các nguồn tin hoặc ký tự diễn giải nội dung.
- rdf:Bag, rdf:Seq, rdf:Alt



Collections



Tóm tắt các dạng cơ bản

- rdf:type
- rdf:Bag, rdf:Seq, rdf:Alt, rdf:_1, ...
- rdf>List, rdf:first, rdf:rest, rdf:nil
- rdf:Statement, rdf:subject, rdf:predicate, rdf:object

RDFa

- Resource Description Framework – in – attributes
- Chuyển RDF vào XHTML pages
- **RDF2RDFa Converter**
- **RDFa Developer** : rdfa.dev.sourceforge.net/

```
...
<body>
<!-- Human-readable content --&gt;
&lt;div&gt;
This is the page content shown to the browser.
Our phone number is &lt;span&gt;+49-89-6004-0&lt;/span&gt;
&lt;/div&gt;
<!-- RDFa Meta-data for machines --&gt;
&lt;div xmlns="http://www.w3.org/1999/xhtml" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" ... &gt;
&lt;span property="vcard:tel" datatype="xsd:string" content="+49-89-6004-0"/&gt;
&lt;/div&gt;
&lt;/body&gt;</pre>
```

Video lectures

- The Semantic Web - An overview

Thực hành

- Thiết kế RDF graph biểu diễn các thông tin về sinh viên và các mối quan hệ với giáo viên, môn học
- Tạo file RDF
- Nghiên cứu chuyển đổi RDF->RDFa
- Tìm hiểu RDFa Developer Tool

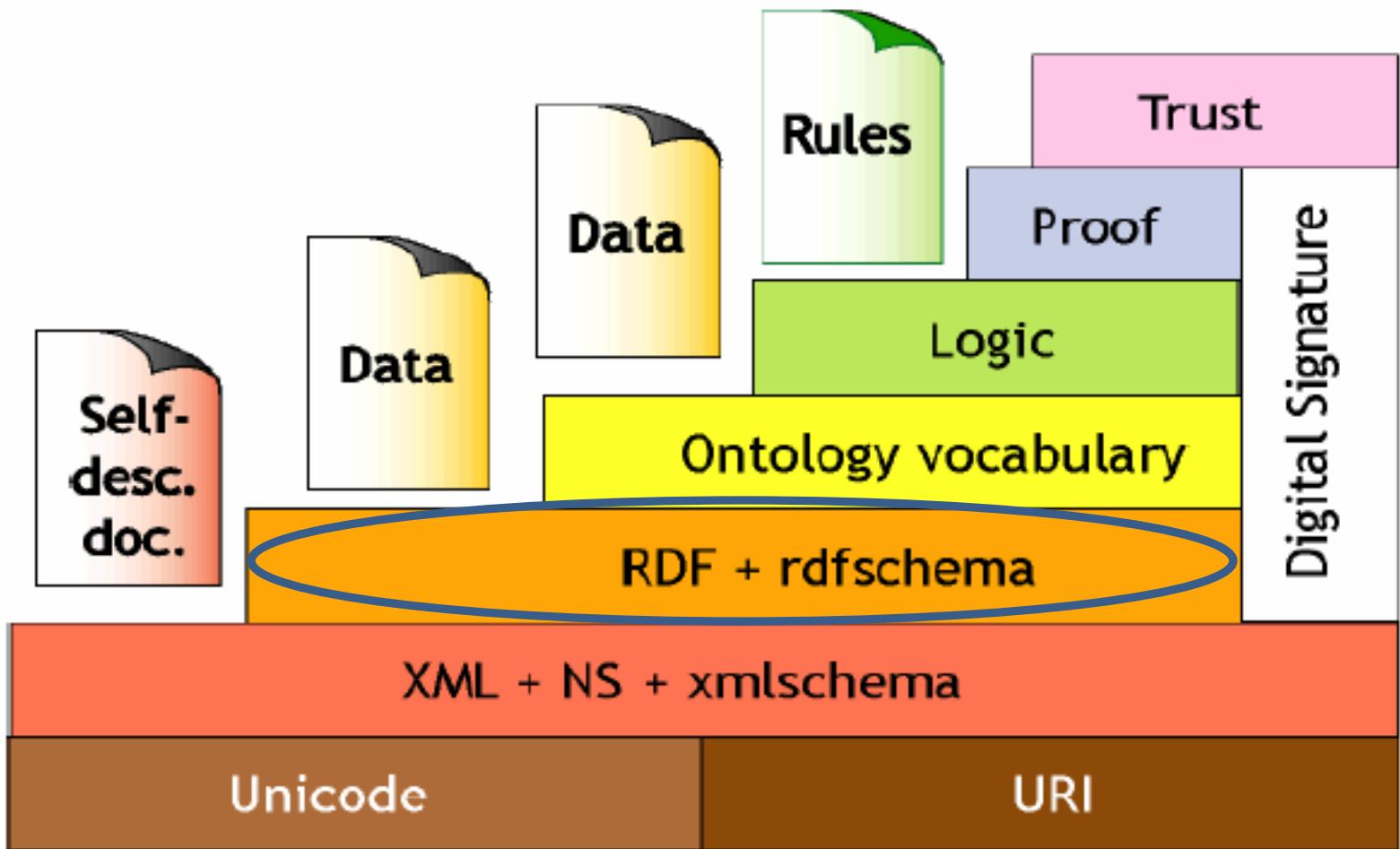
Công nghệ XML và WEB ngữ nghĩa

Khung mô tả dữ liệu RDF

Nội dung chính

- Nhắc lại mô hình cấu trúc web ngữ nghĩa
- Dữ liệu liên kết
- Khái niệm khung mô tả dữ liệu RDF
- Ý tưởng xây dựng RDF dựa trên nền tảng XML

Nhắc lại



Dữ liệu liên kết - Linked Data

- Web ngữ nghĩa không chỉ đơn thuần đưa dữ liệu lên web, vấn đề quan trọng còn là đưa ra các đường link để máy hoặc con người có thể tiếp tục tìm kiếm, truy cập dữ liệu.
- Dữ liệu liên kết giúp mở rộng phạm vi tìm kiếm
- Links cần được hiểu ở nghĩa rộng

Cách thức phát triển dữ liệu liên kết

- Sử dụng URIs thay cho tên gọi
- Sử dụng HTTP URIs để con người có thể tra cứu tên
- Khi tìm kiếm một URI cần cung cấp đầy đủ thêm các thông tin hữu ích thông qua các chuẩn (RDF, SPARQL...)
- Cho thêm đường dẫn vào các URIs để có thể phát hiện thêm thông tin

4 nguyên lý nền tảng

- Tim Berners- Lee tóm tắt 4 nguyên lý nền tảng cho hoạt động của dữ liệu liên kết trong bài viết “Design Issues: Linked Data” (2006):
 - Sử dụng các URIs để xác định, “đặt tên” các “thực thể”
 - Sử dụng giao thức HTTP URI để con người có thể tìm kiếm, duyệt chúng
 - Cung cấp thông tin hữu ích (siêu dữ liệu, mô tả có cấu trúc) về các “thực thể được đặt tên” đó khi URI của chúng được duyệt
 - Chứa các liên kết đến các URIs khác liên quan trong dữ liệu vừa được duyệt giúp có thể duyệt các thông tin khác liên quan.

Web look-up

- Cách đơn giản nhất là trong một file thông tin về đối tượng này sử dụng URI trỏ tới đối tượng khác (kiểu danh sách liên kết)
- Ví dụ trong <http://example.org/smith>
- Có thông tin
- <rdf:Description about="#albert">
 <fam:child rdf:Resource="#brian">
 <fam:child rdf:Resource="#carol">
 </rdf:Description>
- Khi đó có thể truy cập tới Albert
 "http://example.org/smith#albert"

Biến thể - Bạn của bạn

- Friend-Of-A-Friend (foaf)
- foaf:knows [
foaf:mbox <mailto:joe@example.com>;
rdfs:seeAlso <<http://example.com/foaf/joe>>].
- **Đọc là:** Tôi biết người có email
joe@example.com, còn chi tiết thông tin có
thể xem tại địa chỉ
<http://example.com/foaf/joe>

Các chuẩn cho dữ liệu liên kết

- RDF, RDFa, RDF/XML, N3, Turtle

RDF - Resource Description Framework, định dạng dữ liệu cho phép mô tả thực thể, tài nguyên và quan hệ nội tại giữa chúng bằng bộ ba đối tượng – thuộc tính – giá trị (subject – predicate – object).

- **RDFa (RDF – in – attributes)** bổ sung tập các thuộc tính mở rộng cho XHTML để nhúng siêu dữ liệu trong văn bản web.
- **N3 (Notation3)**: cú pháp phi XML của RDF, được thiết kế dễ đọc hơn so với các chú thích RDF/XML.
- **Turtle (Terse RDF Triple Language)** định dạng tuân tự hóa cho các đồ thị RDF, tập con của N3.

RDF & RDFS

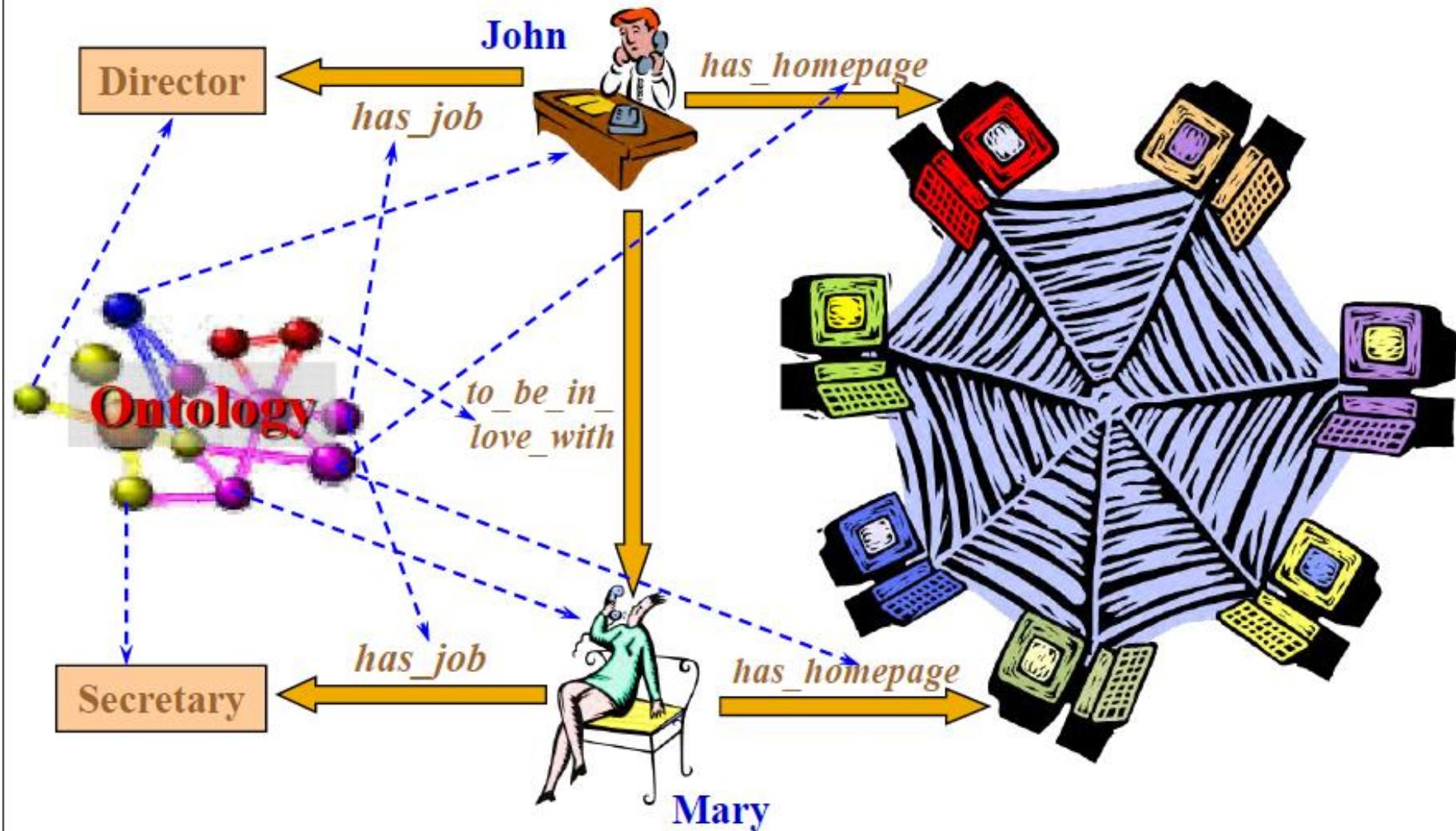
<http://www.w3.org/TR/rdf-primer/>

- RDF = Resource Description Framework
- Địa chỉ tham khảo : <http://www.w3.org/RDF>
- RDF - graphical formalism (+ XML syntax + semantics)
 - Dùng để biểu diễn metadata
 - Mô tả ngữ nghĩa của thông tin theo cách máy tính có thể hiểu
- RDFS = RDF + “schema vocabulary”, ví dụ:
 - Class, Property
 - type, subClassOf, subPropertyOf
 - range, domain

Ví dụ về RDF

- <?xml version="1.0"?>
- <rdf:RDF
- xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:contact="http://www.w3.org/2000/10/swap/pim/contact#">
- <contact:Person
- rdf:about="http://www.w3.org/People/EM/contact#me">
- <contact:fullName>Tran Nguyen Ngoc</contact:fullName>
- <contact:mailbox rdf:resource="mailto:tnn1999@mail.ru"/>
- <contact:personalTitle>Dr.</contact:personalTitle>
- </contact:Person>
- </rdf:RDF>

Minh họa về hoạt động RDF



Mô hình cơ bản của RDF

- Tài nguyên (**Resources**): là tất cả những gì được mô tả bằng biểu thức RDF.
- Thuộc tính (**Properties**): thuộc tính, đặc tính, hoặc quan hệ dùng để mô tả tính chất của tài nguyên.
- Tuyên bố /phát biểu(**Statements**)

Tài nguyên – Resources?

- Tất cả mọi đối tượng được mô tả bởi RDF đều coi là tài nguyên, ví dụ: người, sách, nội dung trang web, các phần tử XML, các đối tượng có thể liên kết từ trang web tới...
- Tài nguyên được định danh bởi *Uniform Resource Identifiers(URI)* - chuỗi ký tự được sắp xếp theo một cú pháp nhất định để nhận dạng các tài nguyên trên web (gồm tài liệu, hình ảnh, tập tin, dịch vụ, hộp thư điện tử, v.v.)
- Hình thức phổ biến nhất của URI là URL (*Uniform Resource Locator*), giúp các chương trình có thể truy cập đến địa chỉ của các tài nguyên một cách đơn giản.

Thuộc tính

- Là các dấu hiệu, đặc điểm của tài nguyên
- Dùng để mô tả tài nguyên: (bài hát) **được biểu diễn bởi...**; (cô gái) **hai mươi tuổi**;...
- Thuộc tính cũng có thể được định danh bởi các URIs

Statements

- Nhằm mục đích khẳng định thuộc tính của tài nguyên
- Cấu trúc một phát biểu bao gồm
- *Subject (đối tượng)*: chính là tài nguyên
- *Predicate (thuộc tính)*: là thuộc tính của tài nguyên
- *Object (giá trị)* : là giá trị của thuộc tính của tài nguyên

Ví dụ về một phát biểu

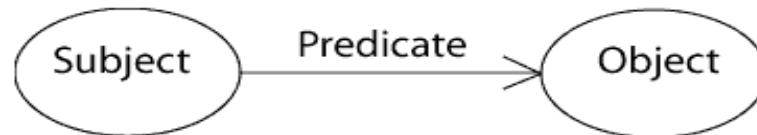
- Trần Nguyên Ngọc giảng dạy Công nghệ XML và Web ngữ nghĩa.

<i>Subject (Resource)</i>	Công nghệ XML và Web ngữ nghĩa
<i>Predicate (Property)</i>	giảng dạy
<i>Object (Literal)</i>	Trần Nguyên Ngọc



Ba cách biểu diễn một Phát biểu

- Dùng bộ ba (RDF triple) SPO (Subject, Predicate, Object)
- Dùng đồ thị (RDF graph): Các node trong đồ thị có thể là các subject và object , cung (arc) trong đồ thi là các predicate. Cung luôn bắt đầu từ **subject** đến **object**



- Dùng XML code

Ví dụ: vẽ liên kết các mệnh đề

- Trần Nguyên Ngọc dạy môn Semantic Web
- Trần Nguyên Ngọc làm việc tại HVKTQS
- Semantic Web là môn học chuyên ngành KHMT
- Trần Nguyên Ngọc có cấp bậc đại úy

Cấu trúc file RDF/XML

- Mô hình RDF thể hiện một mô hình ở mức trừu tượng để định nghĩa metadata (dữ liệu về dữ liệu).
- <?xml version="1.0" encoding="utf-8" ?>
- <rdf:RDF
- xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
- xmlns:extterms="http://www.example.org/terms/">
- <rdf:Description
 rdf:about="http://www.example.org/index.html">
- <extterms:creadtion-date>August 16, 1999</extterms:creadtion-date>
- </rdf:Description>
- </rdf:RDF>

Cấu trúc cơ bản

- [1] RDF ::= ['<rdf:RDF>'] description* ['</rdf:RDF>']
[2] description ::= '<rdf:Description' idAboutAttr? '>' propertyElt* '</rdf:Description>'
[3] idAboutAttr ::= idAttr | aboutAttr
[4] aboutAttr ::= 'about=' URI-reference ""
[5] idAttr ::= 'ID=' IDsymbol ""
[6] propertyElt ::= '<' propName '>' value '</' propName '>'| '<' propName resourceAttr '/>'
[7] propName ::= QName
[8] value ::= description | string
[9] resourceAttr ::= 'resource=' "tham chiếu URI""
[10] QName ::= [NSprefix ':'] name
[11] URI-reference ::= string, interpreted per [URI]
[12] IDsymbol ::= (bất kỳ ID nào hợp lệ nào của XML)
[13] name ::= (bất kỳ tên hợp lệ nào của XML)
[14] NSprefix ::= (bất kỳ tiếp đầu ngữ namespace hợp lệ nào)
[15] string ::= (bất kỳ chuỗi nào)

Khái niệm namespace và qualified name

- **Namespace** là một tập các tên, được định danh bởi các URI, được sử dụng trong các tài liệu XML như các **element type** và **attribute name**. Một **namespace** được khai báo sử dụng một tập các thuộc tính có đã được định nghĩa. Tên của một thuộc tính phải có **xmlns** hay **xmlns:** như là một tiếp đầu ngữ.
- **Qualified name (QName)** bao gồm một tiếp đầu ngữ đã được gán trước đó bởi một URI theo sau là dấu ‘:’ và tên cục bộ.

Ví dụ khai báo Namespace

- NSAttName ::= PrefixedAttName | DefaultAttName
PrefixedAttName ::= 'xmlns:' NCName
DefaultAttName ::= 'xmlns'
NCName ::= (Letter | '_') (NCNameChar)*
NCNameChar ::= Letter | Digit | '.' | '-' | '_' |
CombiningChar | Extender

Ví dụ khai báo QName

- QName ::= (Prefix ':')? LocalPart
Prefix ::= NCName
LocalPart ::= NCName

Thẻ rdf:RDF

- Cho biết rằng nội dung XML tiếp theo dùng để mô tả RDF. Từ khóa này xác định tài liệu này được biểu diễn dưới dạng RDF.
- Tiếp theo là phần khai báo XML namespace được sử dụng trong tài liệu, tùy vào nhu cầu và mục sử dụng mà ta có thể dùng các namespace khác nhau cho từng tài liệu.

Thẻ rdf:Description và rdf:about

- Mô tả subject của phát biểu
- Thẻ bắt đầu *rdf:Description* cho biết bắt đầu mô tả về resource, và tiếp tục định danh resource này dùng thuộc tính *rdf:about* để chỉ ra URI của subject resource.

RDF container

RDF collection

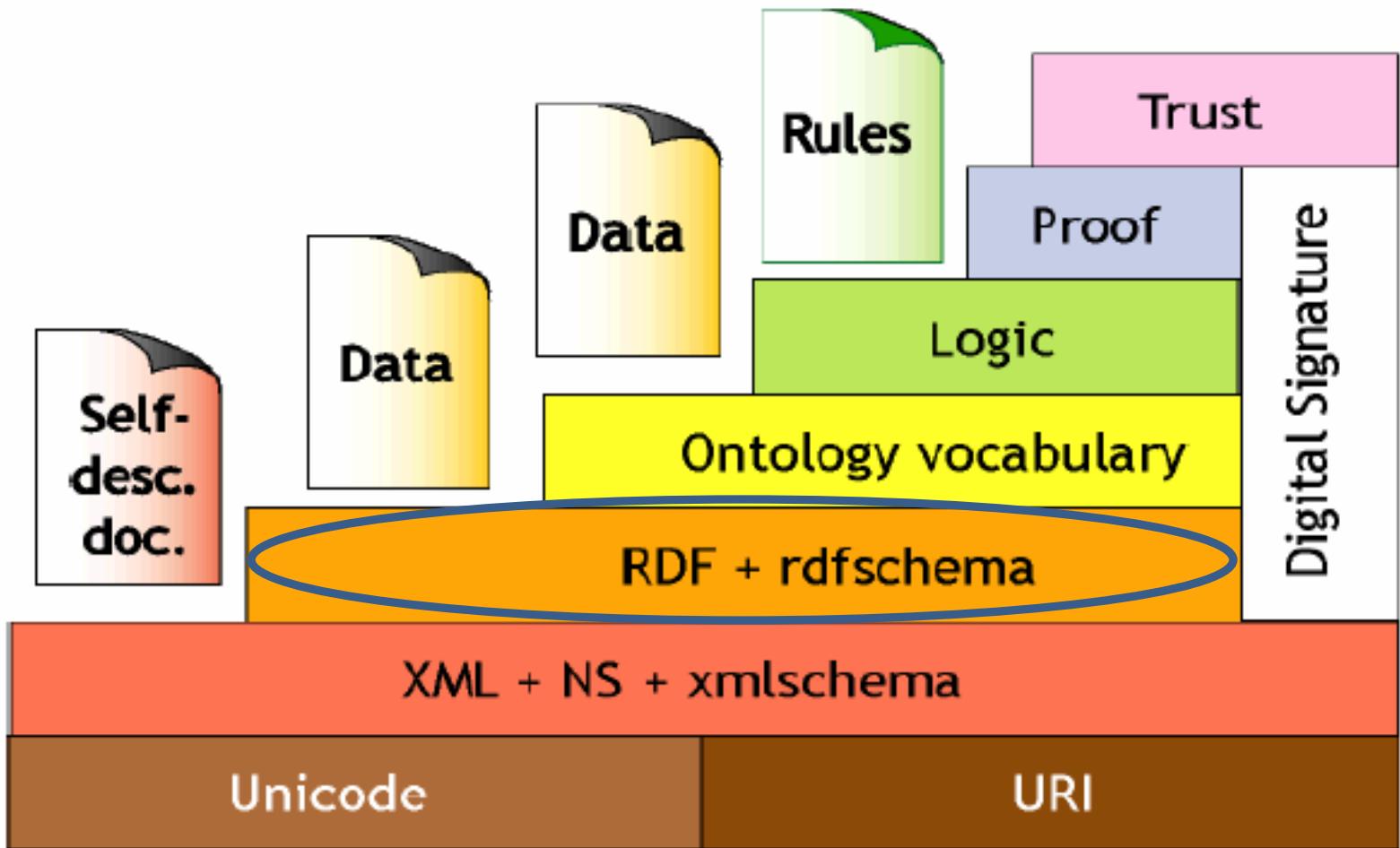
Công nghệ XML và WEB ngữ nghĩa

Khung mô tả dữ liệu RDF

Nội dung chính

- Nhắc lại mô hình cấu trúc web ngữ nghĩa
- Dữ liệu liên kết
- Khái niệm khung mô tả dữ liệu RDF
- Ý tưởng xây dựng RDF dựa trên nền tảng XML

Nhắc lại



Dữ liệu liên kết - Linked Data

- Web ngữ nghĩa không chỉ đơn thuần đưa dữ liệu lên web, vấn đề quan trọng còn là đưa ra các đường link để máy hoặc con người có thể tiếp tục tìm kiếm, truy cập dữ liệu.
- Dữ liệu liên kết giúp mở rộng phạm vi tìm kiếm
- Links cần được hiểu ở nghĩa rộng

Cách thức phát triển dữ liệu liên kết

- Sử dụng URIs thay cho tên gọi
- Sử dụng HTTP URIs để con người có thể tra cứu tên
- Khi tìm kiếm một URI cần cung cấp đầy đủ thêm các thông tin hữu ích thông qua các chuẩn (RDF, SPARQL...)
- Cho thêm đường dẫn vào các URIs để có thể phát hiện thêm thông tin

4 nguyên lý nền tảng

- Tim Berners- Lee tóm tắt 4 nguyên lý nền tảng cho hoạt động của dữ liệu liên kết trong bài viết “Design Issues: Linked Data” (2006):
 - Sử dụng các URIs để xác định, “đặt tên” các “thực thể”
 - Sử dụng giao thức HTTP URI để con người có thể tìm kiếm, duyệt chúng
 - Cung cấp thông tin hữu ích (siêu dữ liệu, mô tả có cấu trúc) về các “thực thể được đặt tên” đó khi URI của chúng được duyệt
 - Chứa các liên kết đến các URIs khác liên quan trong dữ liệu vừa được duyệt giúp có thể duyệt các thông tin khác liên quan.

Web look-up

- Cách đơn giản nhất là trong một file thông tin về đối tượng này sử dụng URI trỏ tới đối tượng khác (kiểu danh sách liên kết)
- Ví dụ trong <http://example.org/smith>
- Có thông tin
- <rdf:Description about="#albert">
 <fam:child rdf:Resource="#brian">
 <fam:child rdf:Resource="#carol">
 </rdf:Description>
- Khi đó có thể truy cập tới Albert
 "http://example.org/smith#albert"

Biến thể - Bạn của bạn

- Friend-Of-A-Friend (foaf)
- foaf:knows [foaf:mbox <mailto:joe@example.com>; rdfs:seeAlso <<http://example.com/foaf/joe>>].
- **Đọc là:** Tôi biết người có email joe@example.com, còn chi tiết thông tin có thể xem tại địa chỉ <http://example.com/foaf/joe>

Các chuẩn cho dữ liệu liên kết

- RDF, RDFa, RDF/XML, N3, Turtle

RDF - Resource Description Framework, định dạng dữ liệu cho phép mô tả thực thể, tài nguyên và quan hệ nội tại giữa chúng bằng bộ ba đối tượng – thuộc tính – giá trị (subject – predicate – object).

- **RDFa (RDF – in – attributes)** bổ sung tập các thuộc tính mở rộng cho XHTML để nhúng siêu dữ liệu trong văn bản web.
- **N3 (Notation3)**: cú pháp phi XML của RDF, được thiết kế dễ đọc hơn so với các chú thích RDF/XML.
- **Turtle (Terse RDF Triple Language)** định dạng tuân tự hóa cho các đồ thị RDF, tập con của N3.

RDF & RDFS

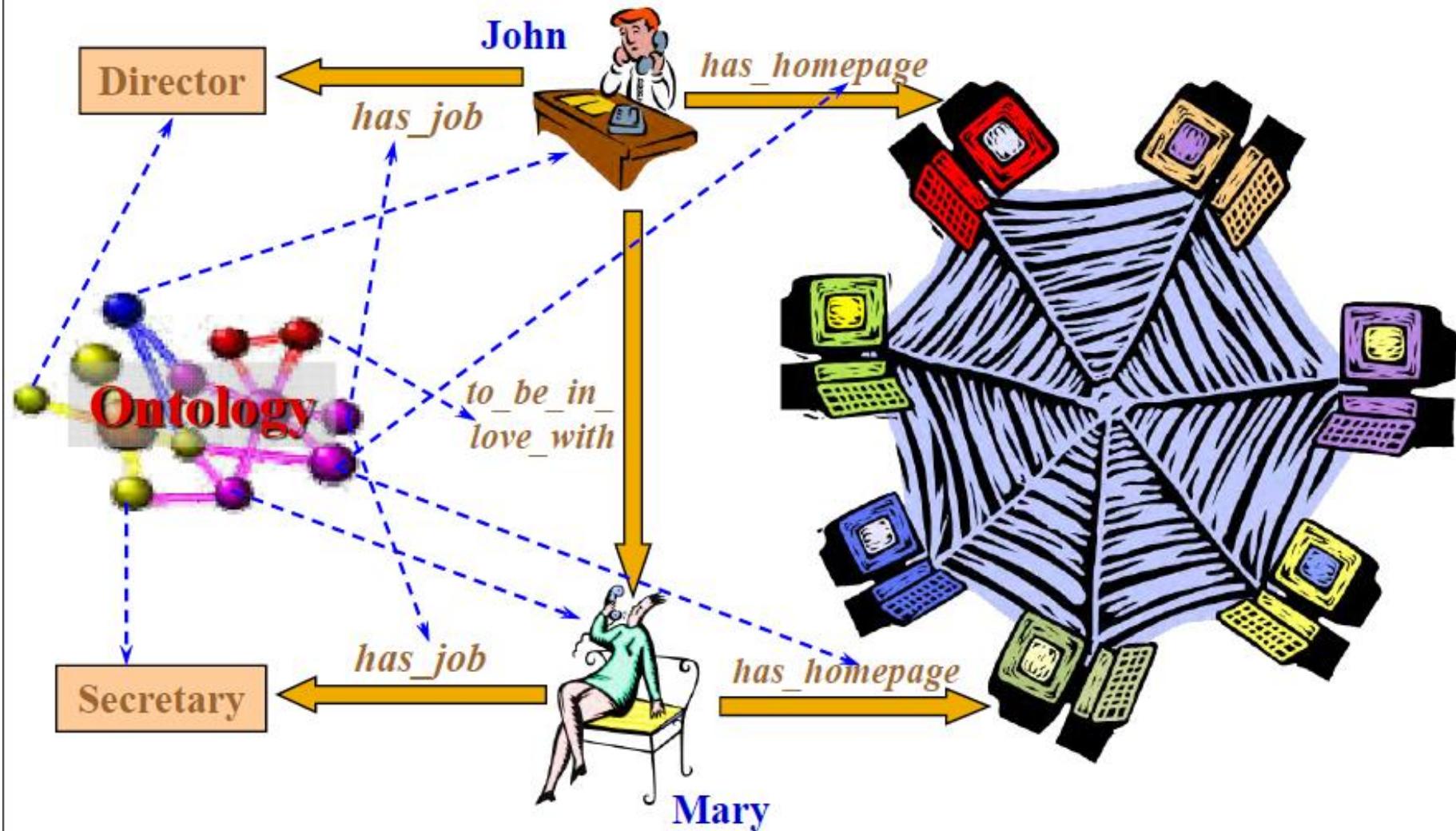
<http://www.w3.org/TR/rdf-primer/>

- RDF = Resource Description Framework
- Địa chỉ tham khảo : <http://www.w3.org/RDF>
- RDF - graphical formalism (+ XML syntax + semantics)
 - Dùng để biểu diễn metadata
 - Mô tả ngữ nghĩa của thông tin theo cách máy tính có thể hiểu
- RDFS = RDF + “schema vocabulary”, ví dụ:
 - Class, Property
 - type, subClassOf, subPropertyOf
 - range, domain

Ví dụ về RDF

- <?xml version="1.0"?>
- <rdf:RDF
- xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:contact="http://www.w3.org/2000/10/swap/pim/contact#">
- <contact:Person
- rdf:about="http://www.w3.org/People/EM/contact#me">
- <contact:fullName>Tran Nguyen Ngoc</contact:fullName>
- <contact:mailbox rdf:resource="mailto:tnn1999@mail.ru"/>
- <contact:personalTitle>Dr.</contact:personalTitle>
- </contact:Person>
- </rdf:RDF>

Minh họa về hoạt động RDF



Mô hình cơ bản của RDF

- Tài nguyên (**Resources**): là tất cả những gì được mô tả bằng biểu thức RDF.
- Thuộc tính (**Properties**): thuộc tính, đặc tính, hoặc quan hệ dùng để mô tả tính chất của tài nguyên.
- Tuyên bố /phát biểu(**Statements**)

Tài nguyên – Resources?

- Tất cả mọi đối tượng được mô tả bởi RDF đều coi là tài nguyên, ví dụ: người, sách, nội dung trang web, các phần tử XML, các đối tượng có thể liên kết từ trang web tới...
- Tài nguyên được định danh bởi *Uniform Resource Identifiers(URI)* - chuỗi ký tự được sắp xếp theo một cú pháp nhất định để nhận dạng các tài nguyên trên web (gồm tài liệu, hình ảnh, tập tin, dịch vụ, hộp thư điện tử, v.v.)
- Hình thức phổ biến nhất của URI là URL (*Uniform Resource Locator*), giúp các chương trình có thể truy cập đến địa chỉ của các tài nguyên một cách đơn giản.

Thuộc tính

- Là các dấu hiệu, đặc điểm của tài nguyên
- Dùng để mô tả tài nguyên: (bài hát) **được biểu diễn bởi...**; (cô gái) **hai mươi tuổi**;...
- Thuộc tính cũng có thể được định danh bởi các URIs

Statements

- Nhằm mục đích khẳng định thuộc tính của tài nguyên
- Cấu trúc một phát biểu bao gồm
- *Subject (đối tượng)*: chính là tài nguyên
- *Predicate (thuộc tính)*: là thuộc tính của tài nguyên
- *Object (giá trị)* : là giá trị của thuộc tính của tài nguyên

Ví dụ về một phát biểu

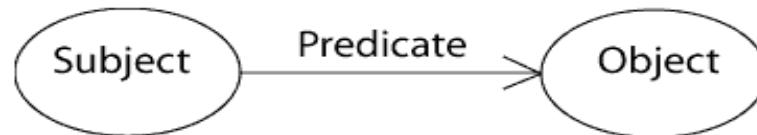
- Trần Nguyên Ngọc giảng dạy Công nghệ XML và Web ngữ nghĩa.

<i>Subject (Resource)</i>	Công nghệ XML và Web ngữ nghĩa
<i>Predicate (Property)</i>	giảng dạy
<i>Object (Literal)</i>	Trần Nguyên Ngọc



Ba cách biểu diễn một Phát biểu

- Dùng bộ ba (RDF triple) SPO (Subject, Predicate, Object)
- Dùng đồ thị (RDF graph): Các node trong đồ thị có thể là các subject và object , cung (arc) trong đồ thi là các predicate. Cung luôn bắt đầu từ **subject** đến **object**



- Dùng XML code

Ví dụ: vẽ liên kết các mệnh đề

- Trần Nguyên Ngọc dạy môn Semantic Web
- Trần Nguyên Ngọc làm việc tại HVKTQS
- Semantic Web là môn học chuyên ngành KHMT
- Trần Nguyên Ngọc có cấp bậc đại úy

Cấu trúc file RDF/XML

- Mô hình RDF thể hiện một mô hình ở mức trừu tượng để định nghĩa metadata (dữ liệu về dữ liệu).
- <?xml version="1.0" encoding="utf-8" ?>
- <rdf:RDF
- xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
- xmlns:extterms="http://www.example.org/terms/">
- <rdf:Description
 rdf:about="http://www.example.org/index.html">
- <extterms:creadtion-date>August 16, 1999</extterms:creadtion-date>
- </rdf:Description>
- </rdf:RDF>

Cấu trúc cơ bản

- [1] RDF ::= ['<rdf:RDF>'] description* ['</rdf:RDF>']
[2] description ::= '<rdf:Description' idAboutAttr? '>' propertyElt* '</rdf:Description>'
[3] idAboutAttr ::= idAttr | aboutAttr
[4] aboutAttr ::= 'about=' URI-reference ""
[5] idAttr ::= 'ID=' IDsymbol ""
[6] propertyElt ::= '<' propName '>' value '</' propName '>'| '<' propName resourceAttr '/>'
[7] propName ::= QName
[8] value ::= description | string
[9] resourceAttr ::= 'resource=' "tham chiếu URI""
[10] QName ::= [NSprefix ':'] name
[11] URI-reference ::= string, interpreted per [URI]
[12] IDsymbol ::= (bất kỳ ID nào hợp lệ nào của XML)
[13] name ::= (bất kỳ tên hợp lệ nào của XML)
[14] NSprefix ::= (bất kỳ tiếp đầu ngữ namespace hợp lệ nào)
[15] string ::= (bất kỳ chuỗi nào)

Khái niệm namespace và qualified name

- **Namespace** là một tập các tên, được định danh bởi các URI, được sử dụng trong các tài liệu XML như các **element type** và **attribute name**. Một **namespace** được khai báo sử dụng một tập các thuộc tính có đã được định nghĩa. Tên của một thuộc tính phải có **xmlns** hay **xmlns:** như là một tiếp đầu ngữ.
- **Qualified name (QName)** bao gồm một tiếp đầu ngữ đã được gán trước đó bởi một URI theo sau là dấu ‘:’ và tên cục bộ.

Ví dụ khai báo Namespace

- NSAttName ::= PrefixedAttName | DefaultAttName
PrefixedAttName ::= 'xmlns:' NCName
DefaultAttName ::= 'xmlns'
NCName ::= (Letter | '_') (NCNameChar)*
NCNameChar ::= Letter | Digit | '.' | '-' | '_' |
CombiningChar | Extender

Ví dụ khai báo QName

- QName ::= (Prefix ':')? LocalPart
Prefix ::= NCName
LocalPart ::= NCName

Thẻ rdf:RDF

- Cho biết rằng nội dung XML tiếp theo dùng để mô tả RDF. Từ khóa này xác định tài liệu này được biểu diễn dưới dạng RDF.
- Tiếp theo là phần khai báo XML namespace được sử dụng trong tài liệu, tùy vào nhu cầu và mục sử dụng mà ta có thể dùng các namespace khác nhau cho từng tài liệu.

Thẻ rdf:Description và rdf:about

- Mô tả subject của phát biểu
- Thẻ bắt đầu *rdf:Description* cho biết bắt đầu mô tả về resource, và tiếp tục định danh resource này dùng thuộc tính *rdf:about* để chỉ ra URI của subject resource.

RDF container

RDF collection

Ngôn ngữ truy vấn SPARQL

Công nghệ XML & Web ngữ nghĩa

Nội dung

- SPARQL là gì
- Giới thiệu một số tính năng chính
- Ví dụ
- Truy vấn dữ liệu RDF

SPARQL là gì?

- Protocol And RDF Query Language
- Phát triển bởi nhóm thuộc W3C: RDF data Access Working Group
- RDF Graph là một tập các triple - bộ ba, mỗi triple bao gồm 3 subject, predecate và object.
- SPARQL là một ngôn ngữ để truy cập thông tin từ các đồ thị RDF
- Ngày 15 tháng 2008, SPARQL đã trở thành một khuyến cáo chính thức của [W3C](#)
- <http://www.slideshare.net/olafhartig/an-introduction-to-sparql>

Các tính năng chính

- Trích thông tin trong các dạng URI
- Trích thông tin từ các đồ thị con
- Xây dựng một đồ thị RDF mới dựa trên thông tin trong đồ thị truy vấn
- (*Trích bài phỏng vấn Tim Berners-Lee*)

A new query language, SPARQL (pronounced "Sparkle"), is designed to make Web pages easier for machines to read, allowing all sorts of different data to be put to work on the Web.

Ví dụ câu truy vấn SPARQL

- Xét một phát biểu RDF triple SPO:
- <<http://example.org/book/book1>>
<<http://purl.org/dc/elements/1.1/title>>
"SPARQL Tutorial"
- Câu truy vấn tìm ra tên quyển sách có cấu trúc gồm mệnh đề SELECT xác định biến chứa kết quả, mệnh đề WHERE mô tả mô hình chứa dữ liệu cần tìm dạng RDF

Ví dụ tìm kiếm đơn giản

- SELECT ?title WHERE {
 <http://example.org/book/book1>
 <http://purl.org/dc/elements/1.1/title> ?title .
}
- Kết quả thu được title= "SPARQL Tutorial"

Ví dụ tìm kiếm nhiều kết quả

- Data

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .  
  
_:a foaf:name "Johnny Lee Outlaw" .  
_:a foaf:mbox <mailto:jlow@example.com> .  
_:b foaf:name "Peter Goodguy" .  
_:b foaf:mbox <mailto:peter@example.org> .  
_:c foaf:mbox <mailto:carol@example.org> .
```

- Query

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>  
SELECT ?name ?mbox  
WHERE  
{ ?x foaf:name ?name .  
?x foaf:mbox ?mbox }
```

- Result

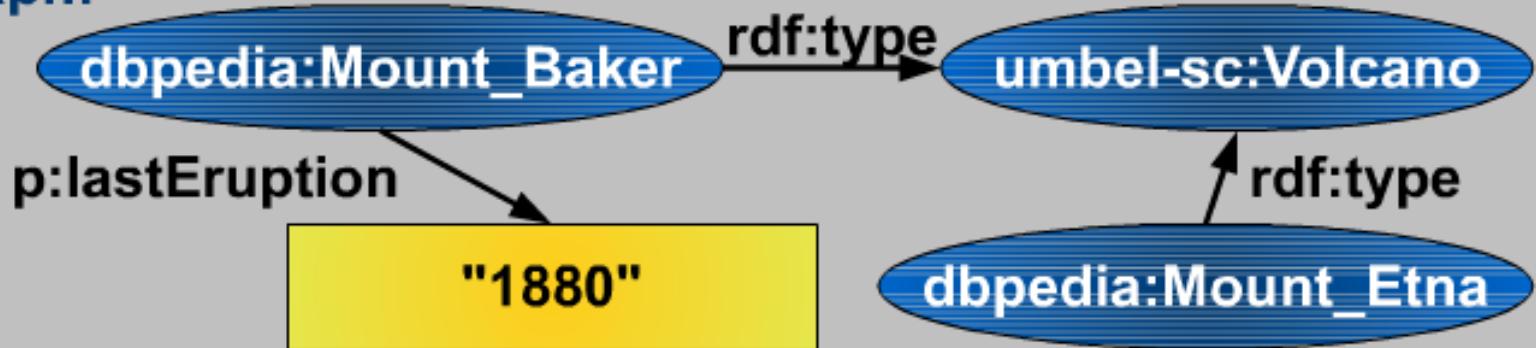
name	mbox
"Johnny Lee Outlaw"	<mailto:jlow@example.com>
"Peter Goodguy"	<mailto:peter@example.org>

Truy vấn dữ liệu RDF với SPARQL

- Ý tưởng chính của các câu truy vấn SPARQL: khớp mẫu (pattern matching) thông qua
- Mô tả các đồ thị con của đồ thị RDF được truy vấn
- Đồ thị con phù hợp với mô tả sẽ là kết quả
- Thực chất: **graph patterns**

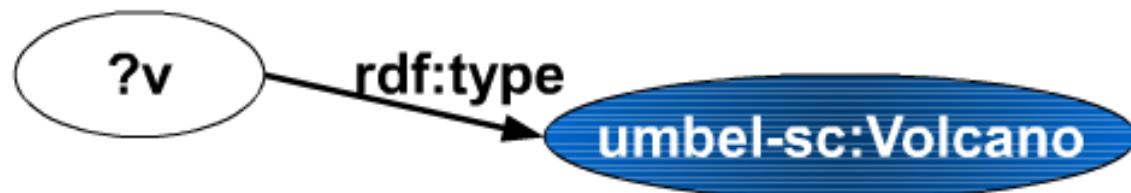
Ví dụ về khớp đồ thị

Queried
graph:



Results:

<code>?v</code>
<code>dbpedia:Mount_Baker</code>
<code>dbpedia:Mount_Etna</code>



Thành phần một câu truy vấn đầy đủ

- Định nghĩa Prefix
- Xác định biến kết quả từ các mệnh đề truy vấn SELECT, DESCRIBE, ASK...
- Phạm vi dữ liệu truy vấn FROM...
- Mẫu truy vấn WHERE...
- Hiệu chỉnh kết quả: ORDER, LIMIT, OFFSET...

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX umbel-sc: <http://umbel.org/umbel/sc/>
SELECT ?v
FROM <http://example.org/myGeoData>
WHERE {
    ?v rdf:type umbel-sc:Volcano .
}
ORDER BY ?name
```

Truy vấn SPARQL cơ bản

- Basic graph pattern (BGP) là tập hợp các bộ ba (triples) được viết theo đúng trình tự (SPO), nếu có nhiều thì phân biệt bởi dấu chấm
- Ví dụ

```
?x foaf:name ?name . ?x foaf:mbox ?mbox
```

Các kiểu truy vấn

- **SELECT:** trích xuất các giá trị thô từ SPARQL endpoint, các kết quả được trả về trong một định dạng bảng.
- **CONSTRUCT:** trích xuất thông tin từ SPARQL endpoint và chuyển kết quả thành dạng RDF hợp lệ
- **ASK:** cung cấp các kết quả dạng True/False đơn giản cho các truy vấn trên SPARQL endpoint
- **DESCRIBE:** trích xuất một đồ thị RDF từ SPARQL endpoint, các nội dung đó được đưa tới endpoint để quyết định dựa trên những thông tin có ích

Ví dụ về công cụ tìm kiếm sử dụng Semantic web và kho dữ liệu RDF

- Swoogle Search Engine, SWSE
- <http://swse.deri.org/>
- <http://swoogle.umbc.edu/>



Thực hành truy vấn dữ liệu với SPARQL

Semantic Web

Tải dữ liệu RDF từ internet

- string uri =
"http://www.mozilla.org/news.rdf";
- MemoryStore store = new MemoryStore();
- store.Import(RdfReader.LoadFromUri(new Uri(uri)));
- Xem lại bài Helloworld để in dữ liệu ra file
hoặc stdout

Tìm hiểu về cách thực hiện câu truy vấn sử dụng thư viện semweb

- Đọc query.html
- Thực hành tạo file chứa dữ liệu
- Thực hành tạo file chứa câu truy vấn
- Truy vấn dữ liệu dạng lệnh
- \$ mono rdfquery.exe

Quản lý dữ liệu truy vấn

- Tạo: Query query;
- query = new SparqlEngine(new StreamReader(queryfile));
- Load dữ liệu
- MemoryStore data = new MemoryStore();
- data.Import(...)
- Tạo kho chứa kết quả truy vấn
- QueryResultSink sink = new SparqlXmlQuerySink(Console.Out);
- Sửa lại thành file thay vì StdOut!
- Truy vấn
- query.Run(data, sink);