Well-formed XML Documents

Assoc. Prof. Dr. Kanda Runapongsa Saikaew
Dept. of Computer Engineering
Faculty of Engineering
Khon Kaen University

Agenda

- Types of XML documents
- Why Well-formed XML Documents
- Rules of Well-formed XML Documents
  - The root element
  - Properly nested elements
  - Quoted attributes
- Entities
- CDATA sections
- Namespaces

Types of XML Documents

- Well-formed documents
  - Well-formed XML documents are easy to process and manage
  - They follow the XML syntax rules but may not have schema
- Valid documents
  - Valid documents are easy to be shared and validated
  - They follow both the XML syntax rules and the rules defined in their schema

XML Document Rules

- XML syntax is defined in the XML specification (http://www.w3.org/TR/REC-xml)
- A parser is a piece of code that reads a document and interpret its contents
- We need to write a well-formed XML document so that the parser will not reject the processing of the document

XML Structure

- Each XML document has both a logical and a physical structure
- Physically, the document is composed of units called entities
- Logically, the document is composed of
  - Declarations
  - Elements
  - Comments
  - Processing instructions

Element and Tags Example

- <name>Thailand</name> is an element
  - <name> is a start tag
  - </name> is an end tag
  - Thailand is an element content
  - name is an element name

Tags

- Similarities of tags in HTML and XML
  - Identify elements

    Example: <table>, <feed>
  - Contain attributes about these elements

    Example:

    <table border="0">

    <feed xmlns="http://www.w3.org/2005/Atom">
- Tags start with the < symbol and end with the > symbol

Empty Element Tag

- If an element is empty, it must be represented either by a start tag followed by an end tag or by an empty-element tag
- Example
  -
</BR>

  (Using a start tag and an end tag)
  -

  (Using an empty-element tag)

Tag Names in XML

- You can start a tag name with a letter, an underscore (_), or a colon (:)
- The next characters may be letters, digits, period (.),
  dash (-), underscore (_), colon (:)
- No tags should begin with any form of "xml"
  - Examples: XML, Xml, XmL
- Tag names are case sensitive
  - Example: <name> != <Name>

Examples of Tag Names

- <1student>
- <superman>
- <computer engineering>
- <xml_is_great>
- <“good”>
- <_wonder>
- <hello,mom>
- <star_wars>
- <jedi&buddha>

Character Data

- Text consists of character data and markup
- In XML definition
  - The text between the start and end tags to be “character data”
  - The text within the tags to be “markup”
  - Example: <name>Thailand</name>
    - “Thailand” is character data
    - “name” is markup

XML Declaration (1/2)

- Indicate that the document is written in XML
- It should be the first line in the document
- An example of an XML declaration

```
<?xml version="1.0" encoding="UTF-8"
   standalone="yes"?>
```

XML Declaration (2/2)

- Three possible attributes in the XML declaration
  - version (required): The XML version.
    - Currently, possible values are “1.0” and “1.1”
  - encoding (optional): The language encoding for the document
    - The default value is UTF-8
  - standalone (optional): Whether the document refers to other documents
    - Set to “yes” if the document does not refer to any external entities
    - Set to “no” otherwise

Elements

- An element represents a logical component of an XML document
- Elements can contain
  - Other elements (sub-elements)
  - Text (character data)
  - The mix of sub-elements and text
- Elements must be properly nested
- Any well-formed XML document needs to have at least one element which is called the root element

Nested Elements Example

- Example tags1

hello

  - Allowed in HTML
  - Not allowed in XML

- Example tags2

hello

  - Properly nested
  - The end tag must be matched with the corresponding start tag

The Root Element

- An XML document must have at least one element which is the root element
- The root element contains all the text and any other elements in the document
- Example: In the sample XML document, the root element is <nation>…</nation>

Attributes

- Descriptive information attached to elements
- Attributes are set inside the start tag of an element
- Attributes are name-value pairs where an attribute value is assigned using an equals sign
- Example: id="th" and version="1.0"

Attribute Names and Values

- Attribute names follow the same rules as tag names
- Attribute values must be assigned and are strings
  - To use them as numbers, we need to translate them
- We must enclose attribute values in quotation marks which can be double and single quotes

Attribute Names and Values Example

•In HTML, it is allowed to write

<table border=0>

…

</table>

•In XHTML (xml-based), it is **not allowed**

to write

<table border=0>

…

</table>

•In XML, attribute values must be quotes with consistent quote type

•This is allowed

<table border="0">

…</table>

•This is allowed

<table border='0'>

…</table>

•This is **not allowed**

<table border="0'>

..

</table>

Elements vs. Attributes

- There can be sub-elements but there is no thing such as a “sub-attribute”
- Each of an element’s attributes may be specified only once, and they may be specified in any order

Elements vs. Attributes Occurrence

- Each element can have multiple occurrence sub-elements

```
<book>
        <chapter>Ch1
        </chapter>
        <chapter>Ch2
        </chapter>
</book>
```

- Each element must have only single occurrence of attributes

```
<book id="b01"
year="2005"/>
```

- We **cannot** have

```
<book chapter="Ch1"
chapter="Ch2"/>
```

Elements vs. Attributes Orders

- Element order is matter
  - –<book>
    <chapter>Ch1
    </chapter>
    <chapter>Ch2
    </chapter></book>

  is different from

  –<book> <chapter>Ch2
  </chapter>
  <chapter>Ch1
  </chapter>
  </book>

- Attributes order is not matter
  - –<book id="b01" year="2005"/>

    is the same as

    –<book year="2005" id="b01"/>

Comments

- Comments are information for the use/author
- <!-- This is a comment -->
- A valid comment should follow these rules
  - The double hyphen '--' must not occur within comments
  - Never place a comment within a tag
  - Never place a comment before the XML declaration

Processing Instructions

- Processing instructions are to represent special instructions for the application using the parser
- All processing instructions, including the XML declaration, start with <? and end with ?>
- Examples
  - <?xml version="1.0" standalone="yes"?>
  - <?xml-stylesheet type="text/xsl" href="nation.xsl"?>

Entities (1/2)

- Entities allow a document to be broken up into multiple storage objects
- They are useful for reusing and maintaining text
- An entity is like a box with a label
  - The label is the entity's name and the content of the box is some sort of text or data

Entities (2/2)

- The entity declaration creates the box and sticks on a label with the name
- There are five predefined XML entities and the users can also define entities themselves in a DTD (Document Type Definition)

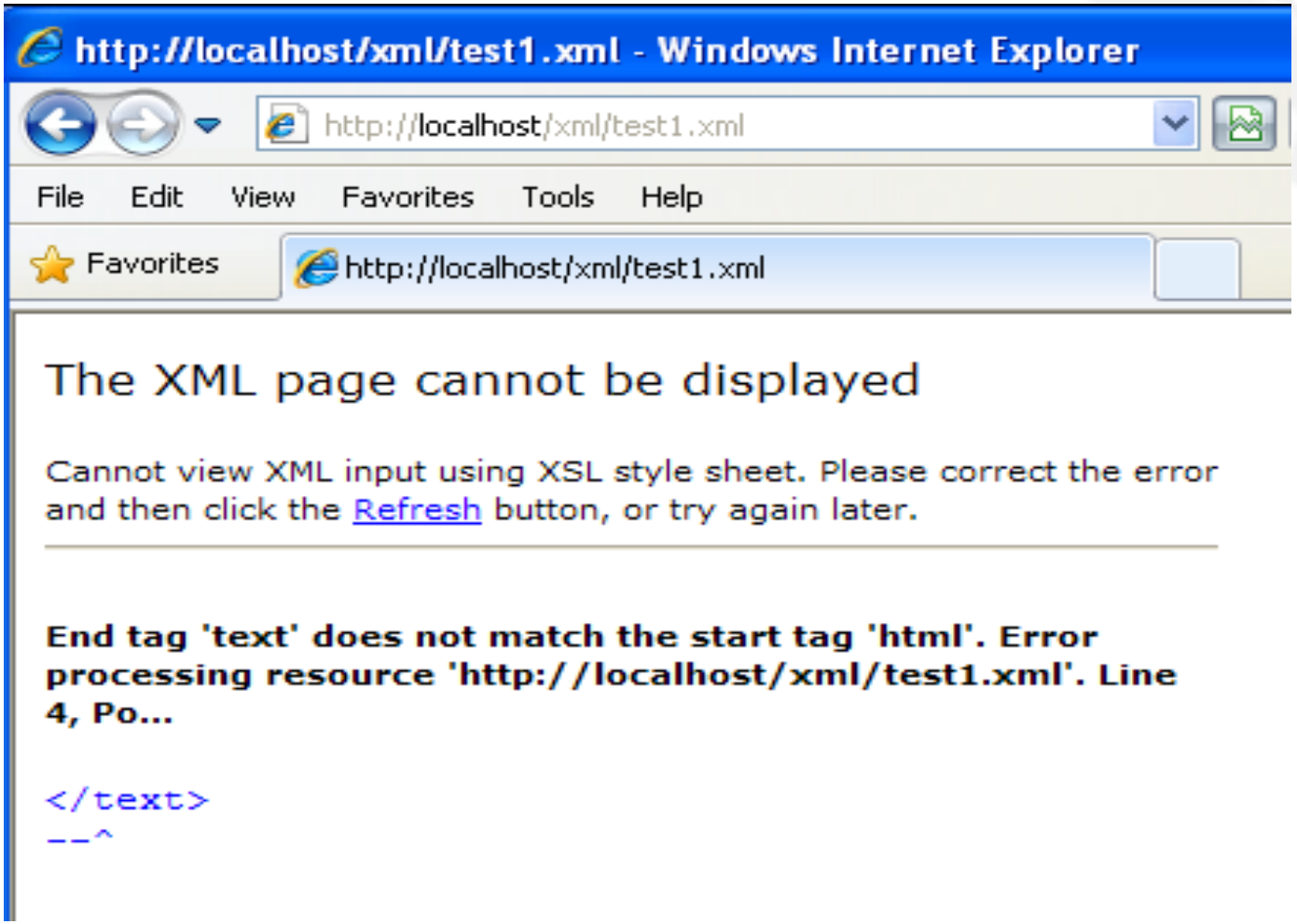
Predefined Entities

- <
  - Produces the left angle bracket <
- >
  - Produces the right angle bracket >
- &
  - Produces the ampersand &
- '
  - Produces a single quote character ‘
- "
  - Produces a double quote character “

Sample XML File with Special Characters

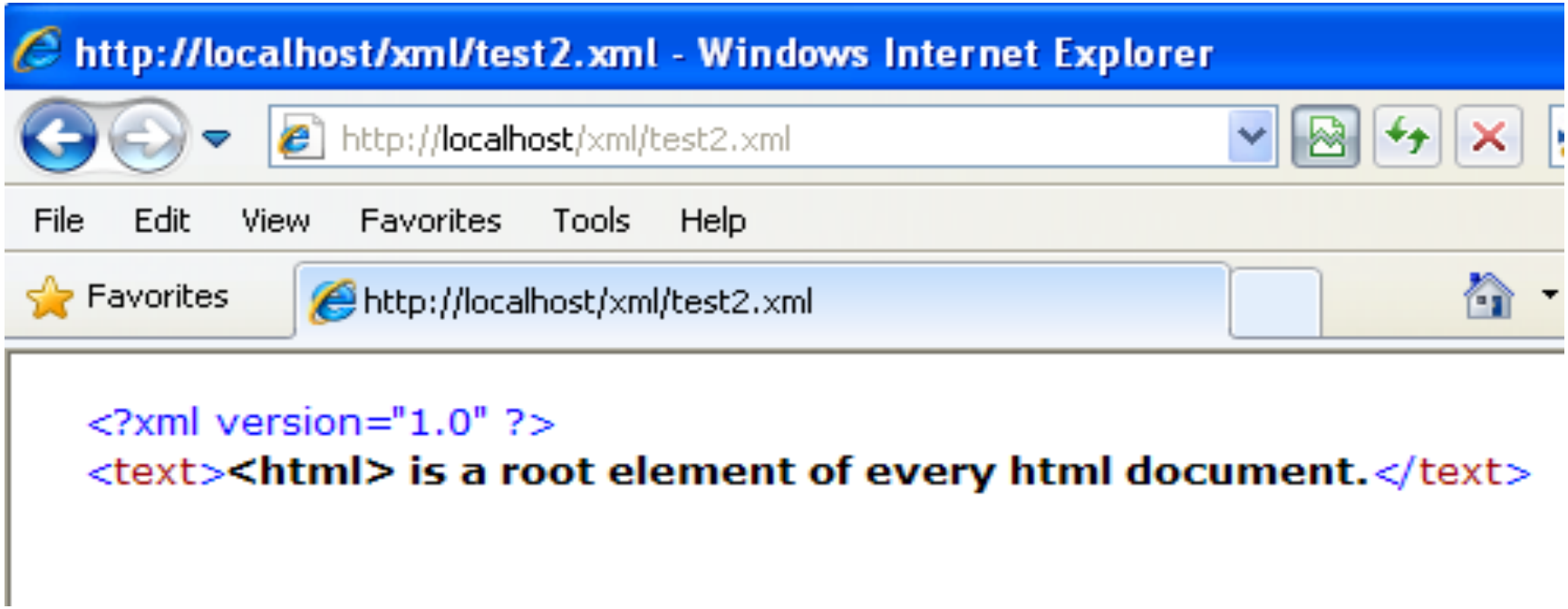
```
<?xml version="1.0"?>
<text>
  <html> is a root element of every html
   document.
</text>
```

XML Document that is Not Well-formed



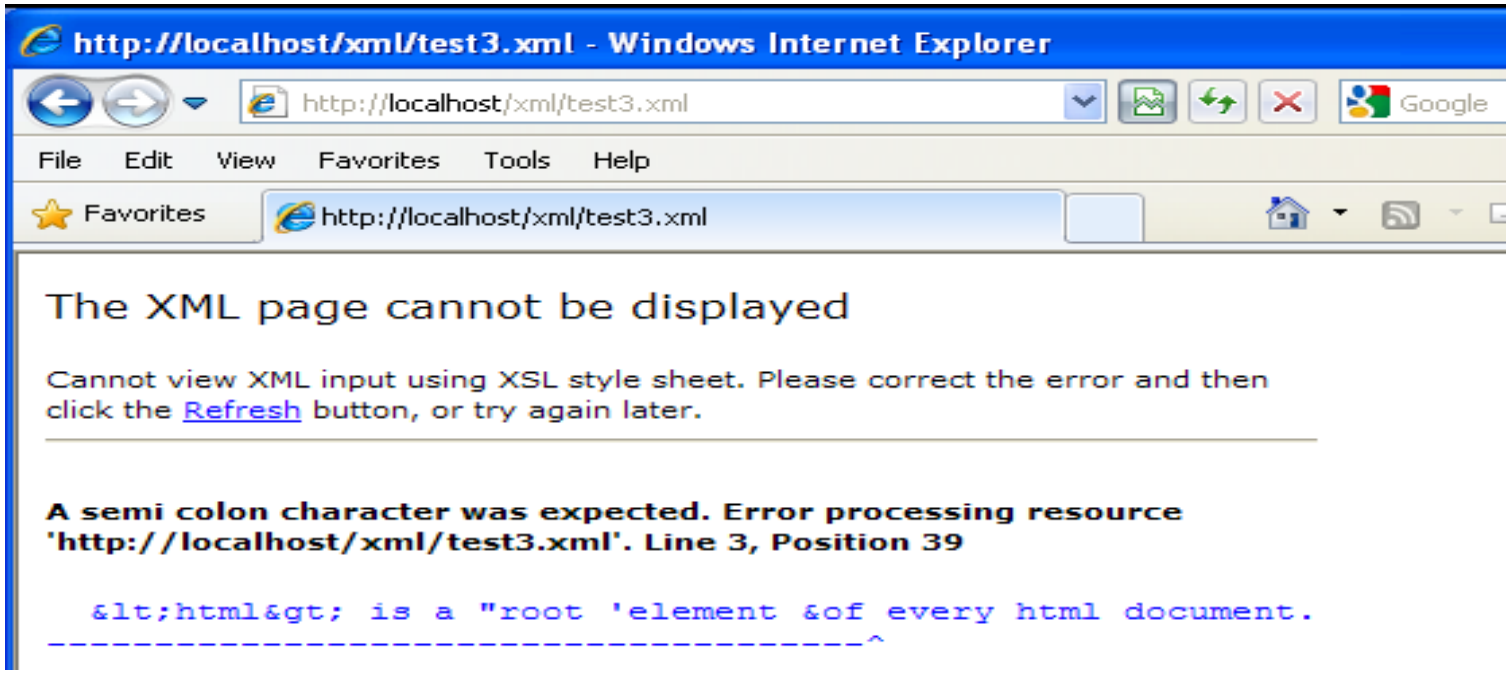
Predefined Entities Example

```
<?xml version="1.0"?>
<text>
    &lt;html&gt; is a root element of every html document.
</text>
```



Testing All Special Characters

```
<?xml version="1.0"?>
<text>
  &lt;html&gt; is a "root 'element &of every html document.
</text>
```



CDATA Sections (1/2)

- CDATA Sections are used to escape blocks of text containing characters which would otherwise be recognized as markup
- All tags and entity references are ignored by an XML processor that treats them just like any character data

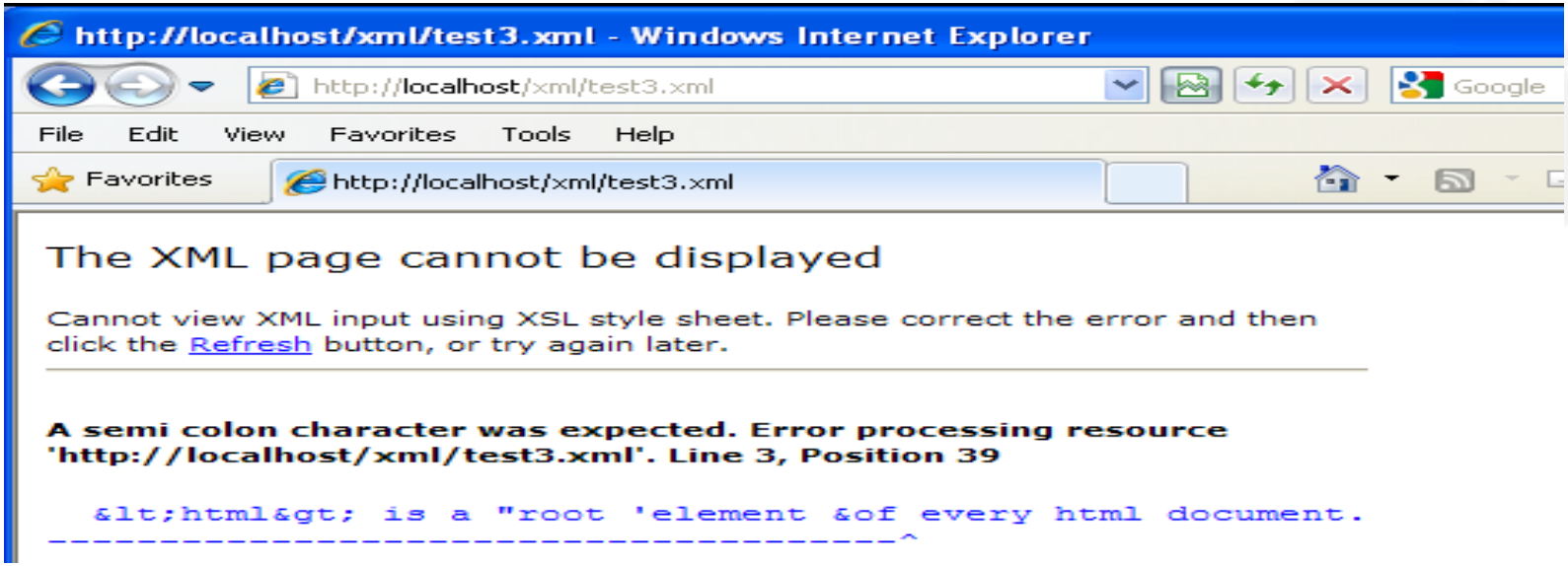
CDATA Sections Examples (1/2)

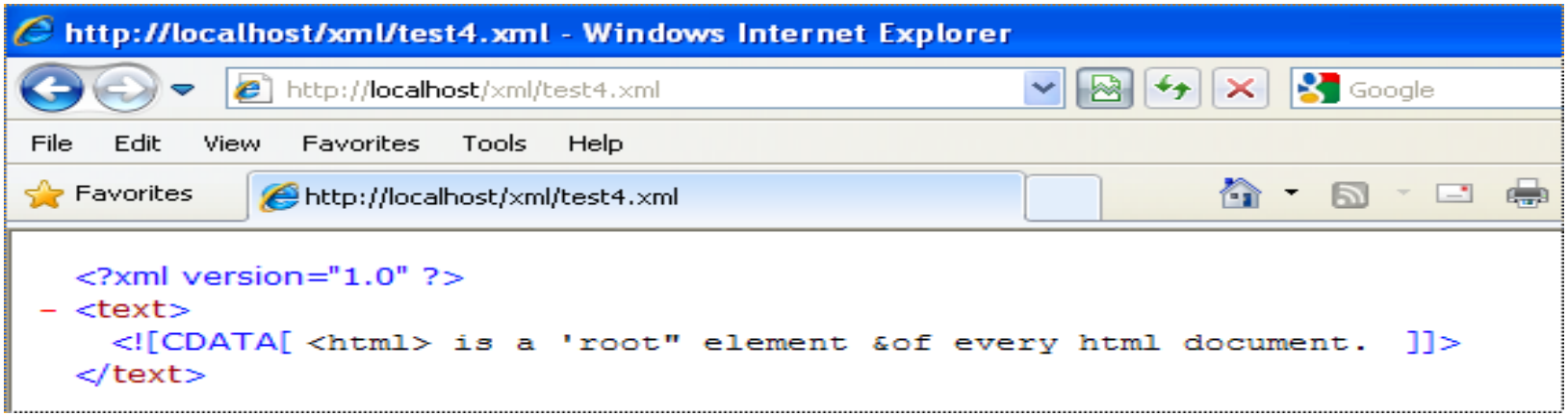
- For example we may want to write
  - <equation>a < 2 = 3</equation>
- The markup for the above equation would be
  - <equation>a < 2 = 3</equation>
  - <equation><![CDATA[a <2 = 3]]></equation>

CDATA Sections Examples (2/2)

```
<?xml version="1.0"?>
<text>
  <![CDATA[<html> is a 'root" element
   &of every html document.]]>
</text>
```

XML File with CDATA Section





Namespaces

- Namespaces allow you to make sure that one set of tags cannot conflict with another
  - Example, ‘title’ of ‘book’ and ‘title’ of ‘page’ in HTML
- Namespaces work by letting you prepend a name followed by a colon to tag and attribute names, changing those names so they do not conflict

Namespaces by Example (1/2)

```
<h:html
   xmlns:xdc="http://abc.com/books"
   xmlns:h="http://www.w3.org/HTML">
        <h:head>
              <h:title>Book Review</h:title>
        </h:head>
        <h:body>
              <xdc:book>
                    <xdc:title>XML</xdc:title>
              </xdc:book>
        </h:body>
</h:html>
```

Namespaces by Example (2/2)

- In the example, the elements prefixed with xdc are associated with a namespace whose name is http://abc.com/books
- Those prefixed with h are associated with a namespace whose name is http://www.w3.org/HTML

Namespace Prefixes

- Prefixes are linked to the full names using the attributes on the top element whose names being xmlns:
- Prefixes are just shorthand placeholders for the full names
- The full names are URLs, i.e., Web addresses

Namespace Defaulting

- We can declare a default namespace and leave out some prefixes
- Anything without a prefix is assumed to be in the default namespace
- A default namespace is considered to apply to the element where it is declared and to all elements with no prefix but it does not apply to attributes

Namespaces Defaulting Example

```
<html
   xmlns:xdc="http://abc.com/books"
   xmlns="http://www.w3.org/HTML">
        <head>
              <title>Book Review</title>
        </head>
        <body>
              <xdc:book>
                    <xdc:title>XML</xdc:title>
              </xdc:book>
        </body>
</html>
```

Attributes and Namespaces

- Attributes can be explicitly assigned to the given namespace
- Attributes without a prefix never belongs to any namespace
- The attributes do not belong to any namespace even if a default namespace is defined for the relevant element

Summary

- A well-formed XML document
  - Must have at least one element
  - Must have one and only one root element
  - Must have elements that are properly nested
  - Must have quotes enclose the attributes
- Namespaces prevent the name conflicts of XML elements

References

- XML standards portal http://www.w3.org/xml
- XML resources
  - http://www.xml.com
  - http://www.oasis-open.org
  - http://www.xml.org
- XML Tutorials
  - http://www-106.ibm.com/developerworks/views/xml/tutorials.jsp
  - http://www.zvon.org
- Sang Shin XML Course Page http://www.javapassion.com/xml/