Institute of
Public Administration

COMP41530 - Web Services in Cloud Computing

Barry Corish
Associate Lecturer, IPA

Lecture 03

Overview

- Review of last week
- The Development Environment
- Practical 01: Build development environment
- Introduction to XML
- Practical 02: Build XML example in Eclipse

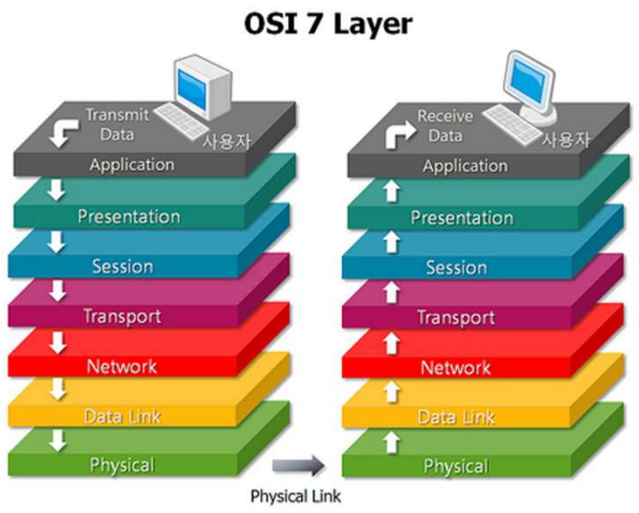
Overview

- **Review of last week**
- The Development Environment
- Practical 01: Build development environment
- Introduction to XML
- Practical 02: Build XML example in Eclipse

Review of last week (1/4)

- Distributed Computing Systems
- Client Server Model
- OSI 7 layer model

OSI 7 Layer Model



Review of last week (2/4)

Internet Protocols
- TCP/IP
- IPv4 Addresses
- DNS
- IPv6
- Ports and Services

Review of last week (3/4)

- Why do “WebServices” use “Web” technologies?
  - Lots of reasons…
- Why do we need middleware?
  - Lots of reasons…

Review of last week (4/4)

- Type of Middleware:
  - Basic
  - Message Oriented Middleware
  - Integration Brokers
  - Implementation: EMS & ESB

TL;DR: WebServices so far…

- So what are we doing with WebServices?
  - We want to send "business" related messages from system to system
  - Take the technologies, network links, tools etc. normally used to send request/deliver Webpages etc..
  - Reuse those things to get our messages and responses from system to system.

Overview

- Review of last week
- **The Development Environment**
- Practical 01: Build development environment
- Introduction to XML
- Practical 02: Build XML example in Eclipse

The Development Environment

- Remember: Not a development course!
  - Don’t panic if you’re not familiar with this
  - One step at a time

The Development Environment

- All programs used are freely downloadable, if not Open Source
- All available as “installers” for:
  - Windows (XP, Vista, 7, 8.x, 10)
  - Mac OSX
  - Linux (various)

The Development Environment

- Here in the IPA, use PC in front of you:
  - All programs downloaded into the Virtual Machine image, install them there
  - Suggest emailing or otherwise sending home work as you go
- Or, if you have a laptop, and prefer to use it:
  - Download and install on that and use both in class at home

The Development Environment

- Get this environment built “at home”:
  - You won’t have enough time to complete practicals and assignments in class.
  - Install programs on your own hardware
  - Get this working now!
    - Don’t wait until just before assignments are due…
  - Make backups as you go.
  - Get eMail working inside VM to get files out of VM!

Applications (1/2)

- Applications we'll be using tonight:
  - Oracle JDK v1.7 (aka Java 7)
    - www.java.com
  - Eclipse JEE
    - www.eclipse.org

Applications (2/2)

- Applications we'll be using later in course:
  - Apache Tomcat
    - www.apache.org
  - SOAPUI
    - www.soapui.org

Overview

- Review of last week
- The Development Environment
- **Practical 01: Build development environment**
- Introduction to XML
- Practical 02: Build XML example in Eclipse

Practical 01

- Build Development Environment
  - Makes a change from the slides!

Overview

- Review of last week
- The Development Environment
- Practical 01: Build development environment
- **Introduction to XML**
- Practical 02: Build XML example in Eclipse

What is XML?

- E<u>x</u>tensible <u>M</u>ark-up <u>L</u>anguage
- A set of rules and formats for putting data into "documents"
- Initially designed for documents
  - Expanded to cover "data"

Why use XML?

- Readable by both Humans and Machines
- Simple and flexible
- Ideal for use over Internet
  - similar to HTML/XHTML and SGML
- Open Standard
- Unicode support
- Widespread adoption and support

What does XML do?

- On it's own, nothing!
  - It's a set of formats for containing and organising data
  - For holding or storing the data
  - For transporting it

Why is it relevant to us?

- Suits SOA
  - Vendor neutral
  - Widespread support
  - Layer of abstraction
  - Ideal for use over networks
- Integral part of WebServices
  - XML is the data format used for WebServices

What other formats are available?

- “WebServices” (according to the strict definition) requires the use of XML
- Strict WebServices are most commonly used in “Enterprise” level systems
- “WebAPI” – essentially “WebServices”, but not following the strictly defined standards.
  - Often use alternative ways of representing data, commonly JSON
- WebAPI/JSON is less formal (and easier?)
- We’ll be using WebServices/XML

XML Files

- Use file extension of ".xml"
- In general, not written by hand!
  - Though you can, and it's easy to read and change once written
- Use descriptive names
  – Should be human readable.

A snippet of XML (incomplete!)

```
<note>
  <to>Jane</to>
  <from>Dave</from>
  <subject>Reminder</subject>
  <body>Remember the milk.</body>
</note>
```

XML Declaration:

- Must start with a declaration of XML type and encoding:

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

- Declaration must be first line.
- Says “this is XML”, the version of XML, and what character set we’re going to use.

Add XML Declaration to our XML

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Jane</to>
  <from>Dave</from>
  <subject>Reminder</subject>
  <body>Remember the milk.</body>
</note>
```

XML Elements

- The basic "data container"
- Can contain any or all of:
  - text
  - other elements
  - attributes

```
<artwork type="sculpture">The
  Kiss<artist>Rodin</artist></artwork>:
```

- There must be a "root element"

XML Root Element Example

```
<?xml version="1.0" encoding="UTF-8"?>
<Customer>
  ...data about the customer...
</Customer>
```

Nesting elements

```
<?xml version="1.0" encoding="UTF-8"?>
<Customer>
  <Address>
     ...data about the customer's address...
  </Address>
  <Account>
     ...data about the customer's account...
  </Account>
</Customer>
```

Don't interleave nested elements

```
<?xml version="1.0" encoding="UTF-8"?>
<Customer>
  <Address>
     ...data about the customer's address...
  <Account>
  </Address>
          ...data about the customer's
  account...
  </Account>
</Customer>
```

XML Attributes

- Individual data items
- Name/Value pairs, added within elements

```
<?xml version="1.0" encoding="UTF-8"?>
<Customer>
  <Address>
     <ContactName>John Smith</ContactName>
     <StreetNumber>14</StreetNumber>
     <StreetName>Accacia Way</StreetName>
     ...etc....
  </Address>
</Customer>
```

Attributes in element start tags (1/3)

- Can also add data about an element to the element start tag
- By convention, used to give more information about the element itself
- Example:

Attributes in element start tags (2/3)

```
<Customer>
  <Address addressType="delivery">
     <ContactName>John Smith</ContactName>
     <StreetNumber>14</StreetNumber>
     <StreetName>Accacia Way</StreetName>
     ...etc....
  </Address>
More…
```

Attributes in element start tags (3/3)

```
  <Address addressType="correspondence">
     <ContactName>John Smith</ContactName>
     <StreetNumber>11</StreetNumber>
     <StreetName>High Street</StreetName>
     ...etc....
  </Address>
</Customer>
```

Example of xml

```
<note>
  <to>Jane</to>
  <from>Dave</from>
  <subject>Reminder</subject>
  <body>Remember the milk.</body>
</note>
```

- Looks like HTML?
  - Comes from the same set of mark-up languages

XML vs. HTML

```
<table>
  <tr>
     <td>Oranges</td>
     <td>Bananas</td>
     <td>Apples</td>
  </tr>
</table>
```

- Focus of HTML is on visual layout

XML vs. HTML

```
<shoppingList>
  <listItem>Oranges</listItem>
  <listItem>Bananas</listItem>
  <listItem>Apples</listItem>
<shoppingList>
```

- Focus of XML is on meaning
  - …but there's lots of crossover between HTML and XML!

What tags are defined in XML?

- In HTML lots:
  - <b>bold</b>
  - <p>paragraph</p>
  - <h1>heading level 1</h1>
  - etc.
- In XML, very few:
  - Define your own to suit the data you want to store

XML is Strict! (1/2)

- Parsers are strict:
  - Badly formed documents will be rejected
  - This is in the standard
- Should use descriptive names
  - Should be human readable

XML is strict! (2/2)

- This is bad HTML, but will display "correctly" most browsers:

```
<p>
   <b>
      Here is some text.
</p>
   </b>
```

  - Most html parsers (browsers) will display fine
- In XML, equivalent will fail!
  - By design and specification

Attributes vs. Elements

- If multiple values required, must use an element
  - Attributes are single valued
- Elements can contain child elements
  - Attributes can't contain anything else

XML Root Element

```
<note>
  <to>Jane</to>
  <from>Dave</from>
  <subject>Reminder</subject>
  <body>Remember the milk!</body>
</note>
```

XML Child Attributes

```
<note>
  <to>Jane</to>
  <from>Dave</from>
  <subject>Reminder</subject>
  <body>Remember the milk!</body>
</note>
```

Child Elements

- Child Elements can contain the same things as root elements
  - Including further "grandchild" elements.
  - Elements can be nested without any nominal limit on depth
  - Tree structure

Special Characters

- Can’t put some characters directly into values, must encode them:
- < and >
  - Use &lt; and &gt; respectively
- & (ampersand)
  - Use &amp;
- ‘ (apostrophe)
  - Use &apos;
- “ (quotation mark)
  - Use &quot;

XML Namespaces (1/2)

- Keeps names used in an XML Document unique
- Traditional to use a URI
  - This is just convention
  - Just a way to get a unique string
- Add an "xmlns" attribute to an element opening tag:
- Example:

XML Namespaces (2/2)

```
<?xml version="1.0" encoding="UTF-8"?>
<Customer
  xmlns="http://www.ipa.ie/SOAandWS/bcorish/customer
  _address">
  <Address>
      <ContactName>John Smith</ContactName>
      <StreetNumber>14</StreetNumber>
      <StreetName>Accacia Way</StreetName>
      ...etc...
  </Address>
</Customer>
```

- If our code "runs into" someone else's code, prevents our "Address" being confused with theirs.

Multiple Namespaces (1/2)

- What if we have several kinds on information in one document?
- Attributes or Elements could have the same names, but mean different things:
- Example:

Namespaces Collision within one document

```
<book>
  <person>
     <fname>Michael</fname>
     <lname>Papazoglou</lname>
     <title>Dr</title>
  </person>
  <title>WebServices &amp; SOA</title>
  <isbn>ABCD1234567</isbn>
</book>
```

Avoid Name Collision with Multiple Namespaces

```
<book>
   xmlns:author="http://www.ipa.ie/library/author"
   xmlns:book="http://www.ipa.ie/library/book" >
   <author:person>
       <author:fname>Michael</fname>
       <author:lname>Papazoglou</lname>
       <author:title>Dr</title>
   </person>
   <book:title>WebServices &amp; SOA</title>
   <book:isbn>ABCD1234567</isbn>
</book>
```

Multiple Namespaces (2/2)

```
<?xml version="1.0" encoding="UTF-8"?>
<Customer
   xmlns:custAddr="http://www.ipa.ie/SOAandWS/customer_address"
   xmlns:acctInfo="http://www.ipa.ie/SOAandWS/accot_info">
   <custAddr:Address>
        <custAddr:ContName>Mary Smith</custAddr:ContName>
        <custAddr:ContTitle>Logistics Manager</custAddr:Cont
   Title>
        ...etc...
   </custAddr:Address>
   <acctInfo:Account>
        <acctInfo:AcctNumber>3533567854</acctInfo:AcctNumber>
        <acctInfo:ContName>Jim Jones</acctInfo:ContName>
        <acctInfo:ContTitle>Financial
   Controller</acctInfo:ConTitle>
        ...etc>
   <acctInfo:/Account>
</Customer>
```

Overview

- Review of last week
- The Development Environment
- Practical 01: Build development environment
- Introduction to XML
- **Practical 02: Build XML example in Eclipse**

Practical 02

- Build XML document in Eclipse IDE

Overview

- Review of last week
- The Development Environment
- Practical 01: Build development environment
- Introduction to XML
- Practical 02: Build XML example in Eclipse

Questions?

?