

## **B336 Advanced Internet Computing**

# **XML Parsing and Processing in Other Programming Languages**

# Learning Objectives

- Get an overview of XML parsers and processors available in other programming languages besides Perl.

# Learning Objectives

- In the scheme of what we are doing in this unit:
  - We are studying how to use XML as an important set of Internet technologies to use as solutions in different areas (for example, the problem defined in Assignment 2).
  - A major part of developing an XML solution is to design and implement software to deal with the information in these XML documents. To do this, we must know how to parse and process XML documents.
  - For us to select the appropriate development environment to use, we should have an idea how much support different languages have for XML parsing and processing.

# Lecture Outline

- Widely-used languages with good XML support
- DOM vs SAX approaches
- Examples of available toolkits

# Programming Languages support for XML in Perl

- In this unit, we are using Perl examples to illustrate what you do with XML parsing and processing. We do so because:
  - Perl has a very simple set of modules for XML parsing, and complete enough for you in your first look at XML.
  - Perl is the most powerful language around for text processing, which at the introductory level should be what you concentrate on in understanding XML structures.
  - The unit coordinator was not able to find time to come up with a comprehensive set of resources, examples and instructions in any other language before the semester starts.

# Programming Languages support for XML in Other Languages

- The above reasons are certain not adequate for you to ignore the available programming XML resources in other languages.
- At the moment, due to the language's popularity, the most comprehensive support for XML exists in JAVA.

# Programming Languages support for XML in Other Languages

- C and C++ also has a large support base for XML
  - arising from the fact that most core system-based processes are still implemented in C/C++, and
  - the popular Expat parser by James Clark is implemented in C.
- For scripting languages besides Perl, Python is enjoying a big surge in popularity as language for XML programming.

# Programming Languages support for XML in Other Languages

- JavaScript is also becoming another popular choice for writing XML parsers and processors
  - Large base of JavaScript programmers who picked up the language through web-page design.
  - Most XML documents are directed towards being served over the web.
  - It is very lightweight - doesn't involve the large resource requirements when starting up and processing.
  - Unfortunately, nowhere near the amount of support in terms of data types and libraries as a "full" programming language.

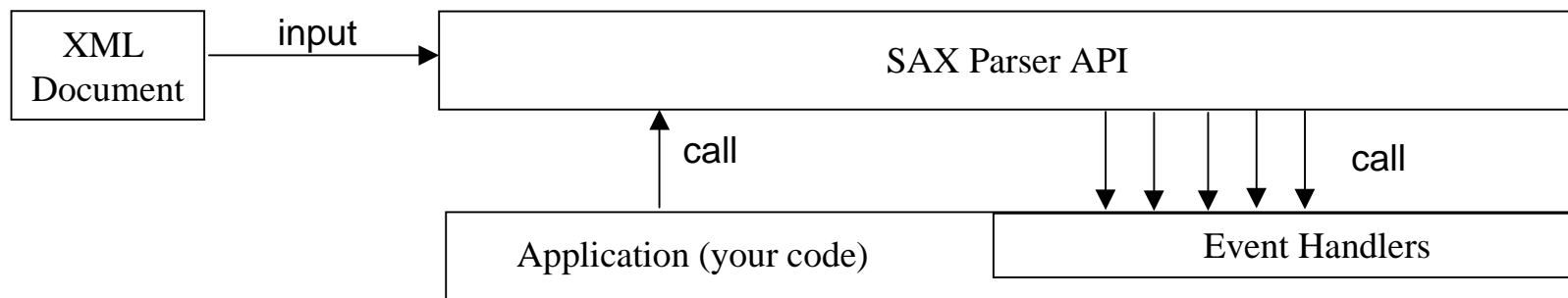
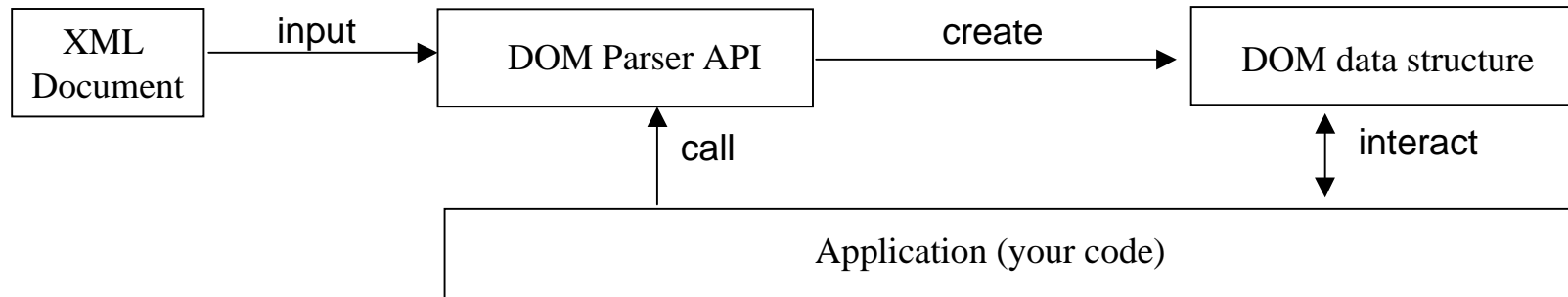
# DOM vs SAX

- In the last lecture, we looked at the parsing approach using DOM.
- There is an approach to parsing XML called SAX (Simple API for XML) which is just as popular as the DOM approach
  - Written by David Megginson (<http://www.megginson.com/SAX>)

# The SAX Approach

- SAX approaches parsing in same vein as Expat (and XML::Parser from last lecture):
  - It provides facilities to define event handlers to handle different parts of the XML document as the parser encounters them.

# DOM vs SAX



# DOM vs SAX

- The DOM approach is good at:
  - Returns a data-structure a programmer to manipulate.
  - Encapsulates all information about the structure of the document, some of which SAX does not return (eg. order of attributes)
- The SAX approach is good at:
  - Efficiency in dealing with large files
    - » does not build a large memory map of the whole document
    - » only parses what it has been instructed to parse,
    - » can begin processing even before the parser finishes reading the whole document - eg, and event handler can start a new thread.
  - Concentrates on the content rather than the layout.
  - When there are a lot of external API calls (eg database specific system calls) your event handlers can make to deal with the data.

# A question of efficiency

- The SAX approach is an important part of XML parsing due to efficiency.
  - It is just too inefficient using the DOM approach to parse documents which are very large, especially when we only want a small part of the data.
- DOM on the other hand is also important due to the data and object-centric way we approach most programming today.
  - We need to have adequate facilities to manipulate data.

# Popular XML Parsing Toolkits today

- Those which have long history behind them, which many new XML software libraries are built upon:
  - Expat by James Clark ([www.jclark.com/xml/expat.html](http://www.jclark.com/xml/expat.html))
  - SAX - not defined in any language, and has implementations in Perl, Java, C/C++ and many other languages.
  - XP, also by James Clark ([www.jclark.com/xml/xp/index.html](http://www.jclark.com/xml/xp/index.html))
  - Lark by Tim Bray, in Java ([www.textuality.com/Lark/](http://www.textuality.com/Lark/))
  - LT XML - by Language Technology Group at U. of Edinburg ([www.ltg.ed.ac.uk/software/xml/](http://www.ltg.ed.ac.uk/software/xml/))
  - XML::Parser by Larry Wall (see Perl documentation)
  - SXP - Silfide XML Parser ([www.loria.fr/projets/XSilfide/EN/sxp/](http://www.loria.fr/projets/XSilfide/EN/sxp/))

# Popular XML Parsing Toolkits today

- Widely used today:
  - XML for Java (XML4J) by IBM AlphaWorks - widely used and conforms well to W3C standards ([www.alphaworks.ibm.com/tech/xml4j](http://www.alphaworks.ibm.com/tech/xml4j))
  - Microsoft XML Parser (MSXML) - implemented as a COM component ([msdn.microsoft.com/xml/general/xmlparser.asp](http://msdn.microsoft.com/xml/general/xmlparser.asp))
  - Ælfred in Java- concentrates on optimising speed and size, especially good to use in an applet ([http://www.opentext.com/services/content\\_management\\_services/xml\\_sgmml\\_solutions.html#aelfred\\_and\\_sax](http://www.opentext.com/services/content_management_services/xml_sgmml_solutions.html#aelfred_and_sax))
  - Java Standard Extensions for XML by Sun Microsystems ([java.sun.com/products/xml](http://java.sun.com/products/xml))
  - Perl XML modules (<http://cpan.valueclick.com/modules/by-module/XML/>)
  - Python parser ([www.python.org/topics/xml](http://www.python.org/topics/xml))
  - ... and many, many more.

# What Language Should You Choose?

- Depends on trade-offs between:
  - Whether the languages and their libraries have support for features which will enhance your particular solutions.
  - What languages you and your project team are comfortable with.
  - What language the developers in the problem area is mostly using - for integration.
  - What resources (especially software) your organization own, have access to, and is planning to obtain - again for integration..
- At the moment, any facilities available for XML you will find some implementation in JAVA... but is that the best choice based on the criteria above.

# Reference

- For a good list of the most popular XML parsers and programming support today, refer to:
  - [http://www.xml.com/pub/rg/XML\\_Parsers](http://www.xml.com/pub/rg/XML_Parsers)

# In the next lecture...

- Look at example of XML is applied to different areas, and what designing an XML solution involves.
- Next week, we will look at the prevalent XML developments (eg. Microsoft's .NET strategy, Oracle's database integration packages, open source projects such as Apache-XML) happening on the Internet today.