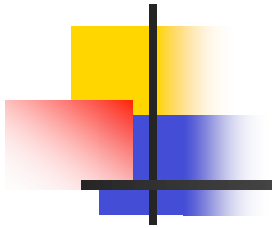
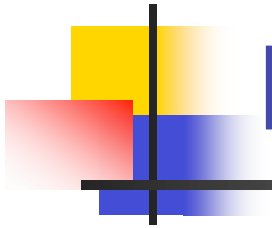


B336 Internet Systems Programming

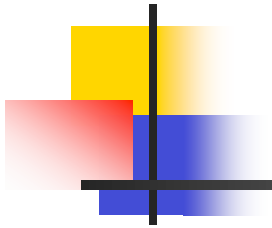
XML Parsing and Processing in Other Programming Languages

(Week 8 Lecture 1 Part 2)



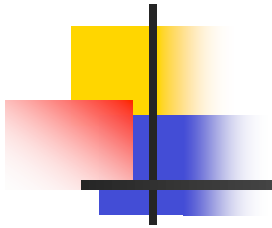
Learning Objectives

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



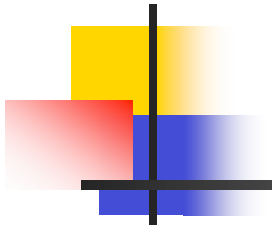
Learning Objectives

- 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.
  - 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.



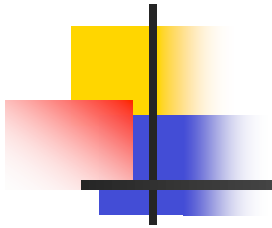
Lecture Outline

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



Programming Languages support for XML in Perl

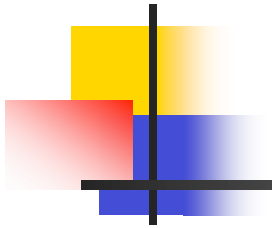
- In this unit, we are using Perl examples to illustrate what you do with XML parsing and processing. We do so because:
  - Perl has a very simple set of modules for XML parsing, and complete enough for you in your first look at XML.
  - Perl is the most powerful language around for text processing, which at the introductory level should be what you concentrate on in understanding XML structures.



Programming Languages support for XML in Other Languages

- The above reasons does not mean we ignore available support for XML in other programming languages.

- There are more comprehensive support for XML in other languages like Java and Microsoft's new C# language.



JAVA's support for XML

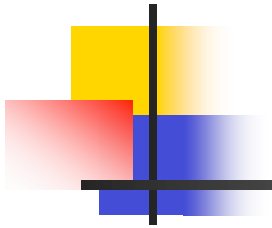
- Some example Java XML APIs:
  - Sun's Java APIs for XML (JAX)
  - Xerces parser from Apache XML Project
  - IBM's XML4J
- See other examples at:
  - http://www.xml.com/pub/rg/Java
  - http://www.xml.com/pub/rg/Java_Parsers

Example code from JAX

```
DocumentBuilderFactory factory =
                DocumentBuilderFactory.newInstance();
DocumentBuilder builder = factory.newDocumentBuilder();
Document document = builder.parse("priceList.xml");

NodeList list = document.getElementsByTagName("name");
Node thisNode = list.item(0); // loop through list
Node thisChild = thisNode.getChildNode();
if (thisNode.getFirstChild() instanceof org.w3c.dom.TextNode) {
   String data = thisNode.getFirstChild().getData();
}

if (data.equals("Mocha Java")) {
   // new node will be inserted before Mocha Java
   Node newNode = document.createElement("coffee");
   Node nameNode = document.createElement("name");
   TextNode textNode = document.createTextNode("Kona");
   nameNode.appendChild(textNode);

   Node priceNode = document.createElement("price");
   TextNode tpNode = document.createTextNode("13.50");
   priceNode.appendChild(tpNode);

   newNode.appendChild(nameNode);
   newNode.appendChild(priceNode);
   thisNode.insertBefore(newNode, thisNode);
}
```

- Note that the method names are the same as with the XML::DOM API in Perl. These names are defined in DOM, and are therefore language independent.

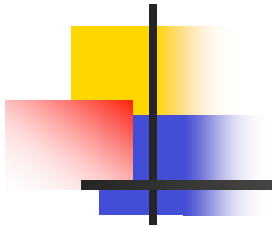
Source: http://java.sun.com/webservices/docs/ea2/tutorial/doc/IntroWS5.html#63986



Programming Languages support for XML in Other Languages

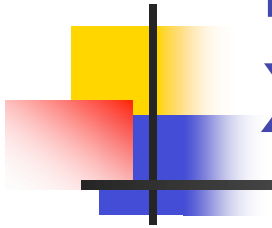
- C and C++ also has a large support base for XML
  - arising from the fact that most core system-based processes are still implemented in C/C++, and
  - the popular Expat parser by James Clark is implemented in C.

- For scripting languages besides Perl, Python is enjoying a big surge in popularity as language for XML programming.



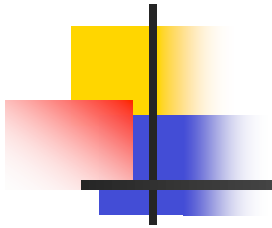
Programming Languages support for XML in Other Languages

- JavaScript and even CSS are 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 once a browser is started
    - 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.



Programming Languages support for XML in Other Languages

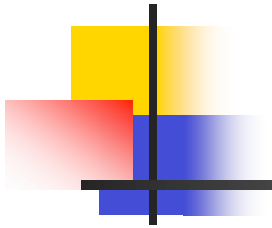
- All of the newer languages, such as C# and PHP, all have major libraries supporting XML.



DOM vs SAX

- In the last lecture, we looked at the parsing approach using DOM.

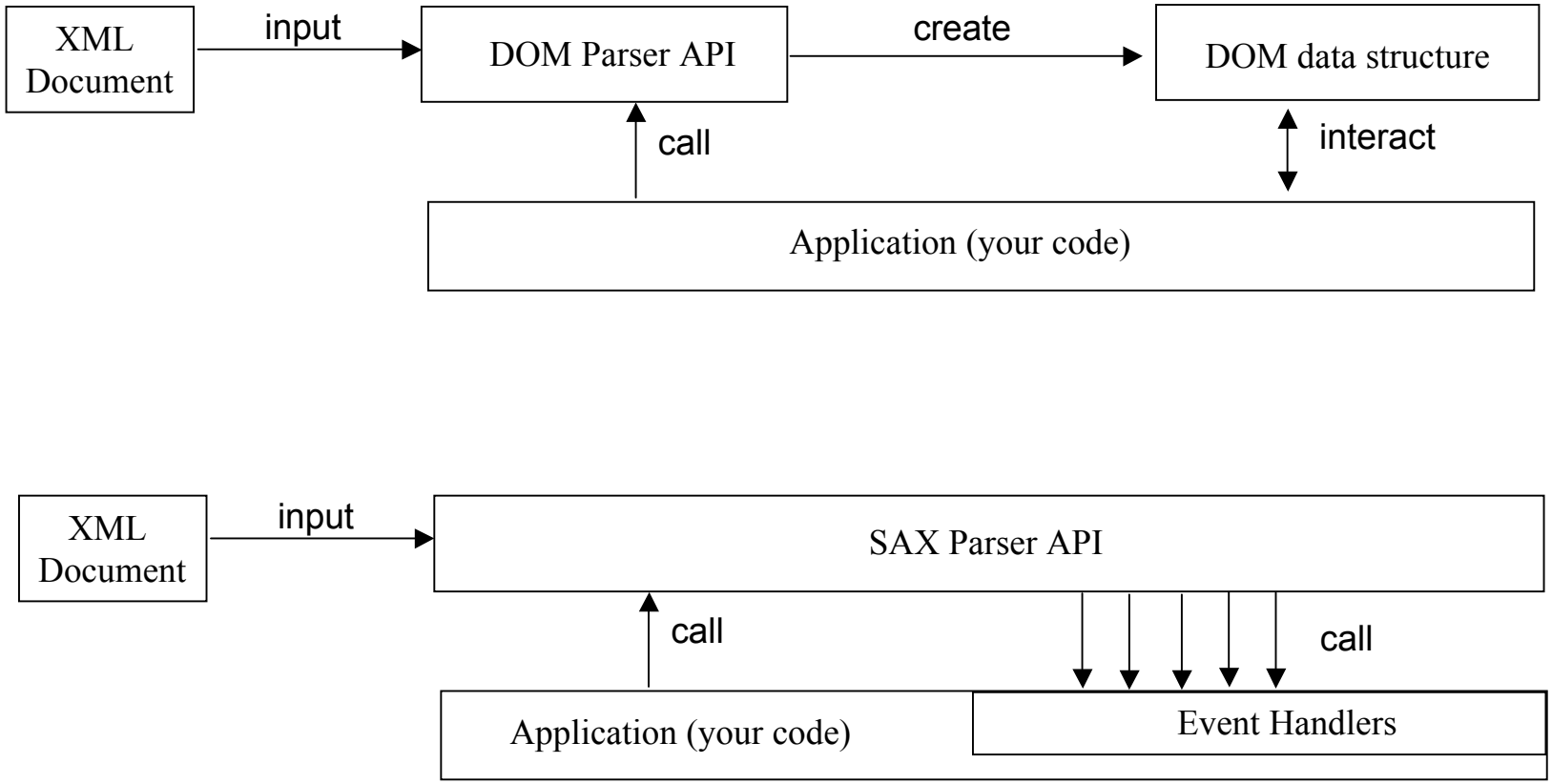
- There is an approach to parsing XML called SAX (Simple API for XML) which is just as popular as the DOM approach
  - Written by David Megginson (http://www.megginson.com/SAX)

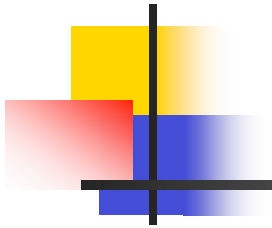


The SAX Approach

- 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.

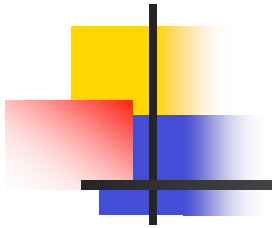
DOM vs SAX


XML Document
input
DOM Parser API
create
DOM data structure
call
interact
Application (your code)
XML Document
input
SAX Parser API
call
call
Application (your code)
Event Handlers



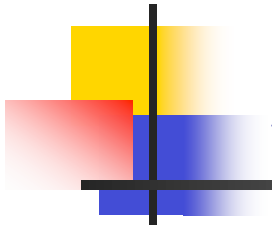
DOM vs SAX

- 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)



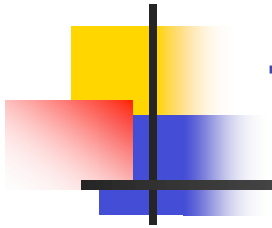
DOM vs SAX

- 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.



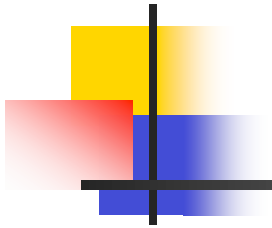
A question of efficiency

- 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 a 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.



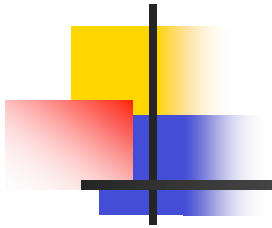
Popular XML Parsing Toolkits today

- 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)
  - 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)
  - Lark by Tim Bray, in Java (www.textuality.com/Lark/)
  - LT XML - by Language Technology Group at U. of Edinburg (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/



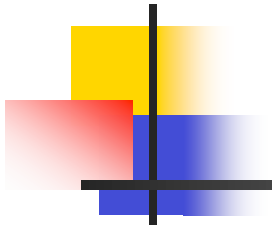
Popular XML Parsing Toolkits today

- Widely used today:
  - XML for Java (XML4J) by IBM AlphaWorks - widely used and conforms well to W3C standards (www.alphaworks.ibm.com/tech/xml4j)
  - Microsoft XML Parser (MSXML) - implemented as a COM component (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_sgml_solutions.html#aelfred_and_sax)
  - Java Standard Extensions for XML by Sun Microsystems (java.sun.com/products/xml)
  - Perl XML modules (http://cpan.valueclick.com/modules/by-module/XML/)
  - Python parser (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.



Reference

- Read chapters 16, 19, 20 an 21 for a flavour of XML support in different languages.
  - You are not required to understand a lot of the technical details in these chapters (the descriptions are too brief to be useful), but just to get an appreciation of the variety available.
- For a list of the popular XML parsers and programming support today, refer to:
  - http://www.xml.com/pub/rg/XML_Parsers