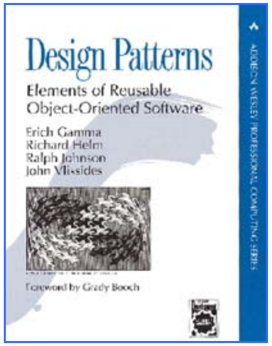
CSC207H: Software Design
Lecture 7

Wael Aboelsaadat

wael@cs.toronto.edu
http://ccnet.utoronto.ca/20075/csc207h1y/
Office: BA 4261
Office hours: R 5-7

Acknowledgement: These slides are based on material by Prof. Karen Reid

Design Pattern Space



Design Patterns

Elements of Reusable Object-Oriented Software

Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides

Foreword by Grady Booch

ADDISON WESLEY PROFESSIONAL COMPUTING SERIES

Creational	Structural	Behavioral
Factory Method Abstract Factory Builder Prototype Singleton	Adapter Bridge Composite Decorator Flyweight Facade Proxy	Interpreter Template Method Chain of Responsibility Command Iterator Mediator Memento Observer State Strategy Visitor

Pattern: Template Method

What’s Wrong With This? ☺

```
public class PizzaMaker {

   public void cookPizzas(List pizzas) {
     for (int i=0; i<pizzas.size(); ++i) {
        Object pizza = pizzas.get(i);

        if (pizza instanceof ThinCrustPizza) {
           ((ThinCrustPizza)pizza).cookInWoodFireOven();
        }
        else
            if (pizza instanceof PanPizza) {
              ((PanPizza)pizza).cookInGreasyPan();
             }
             else {

             }
       }
```

The Open-Closed Principle

- *Classes should be open for extension, but closed for modification*
  - I.e., you should be able to extend a system *without* modifying the existing code

- The type-switch in the example violates this
  - Have to edit the code every time the marketing department comes up with a new kind of pizza

Abstraction is the Solution

- Solve the problem by creating a `Pizza` interface with a `cook` method
  - Or an abstract base class whose cook method must be overridden by every child

- Simple, right?

How Open Should You Be?

```
public abstract class Pizza {
  public final void cook() {
    placeOnCookingSurface();
    placeInCookingDevice();
    int cookTime = getCookTime();
    letItCook(cookTime);
    removeFromCookingDevice();
  }
  protected abstract void placeOnCookingSurface();
  protected abstract void placeInCookingDevice();
  protected abstract int  getCookTime();
  protected abstract void letItCook(int min);
  protected abstract void removeFromCookingDevice();
}
```

Template Method Design Pattern

- The *Template Method* design pattern is used to set up the **skeleton** of an algorithm
  - Details then filled in by concrete subclasses

- But what if someone wants to do something you didn't anticipate?
  - E.g., wants to add a `PancakePizza` that has to be flipped over halfway through the cooking process

Override the Template Method?

```
public final void cook() {
  placeOnCookingSurface();
  placeInCookingDevice();
  int cookTime = getCookTime();
  letItCook(cookTime/2);
  flip();
  letItCook(cookTime/2);
  removeFromCookingDevice();
}
```

- But `cook` was `final`
- And it’s storing up trouble for the future

Squeeze It Somewhere Else?

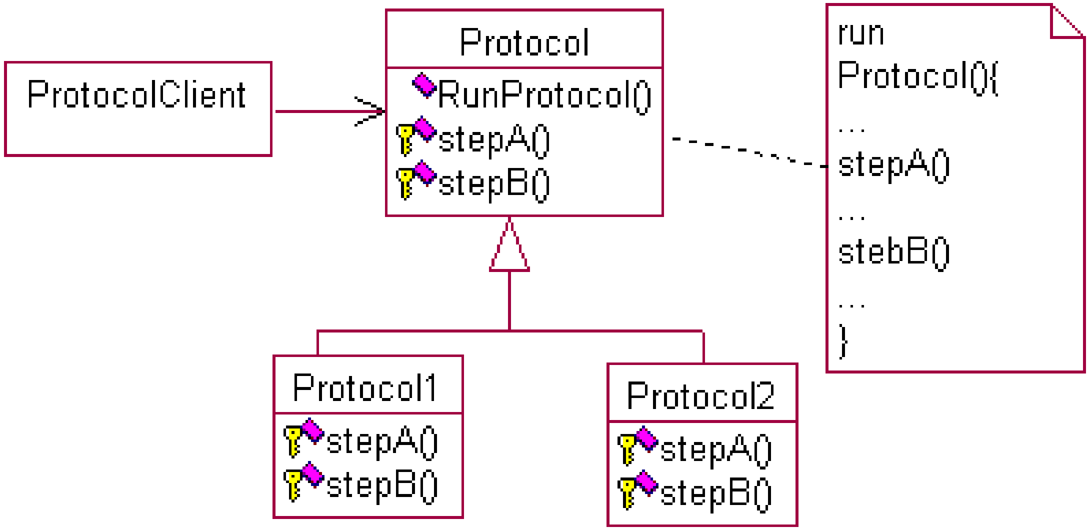
```
protected void removeFromCookingDevice() {
  flip();
  letItCook(cookTime);
  …remove from skillet…
}
```

- `removeFromCookingDevice` shouldn’t be doing other things
  - Think about the documentation
- And once again, we’re storing up trouble for the future

Leave Space for Future Growth?

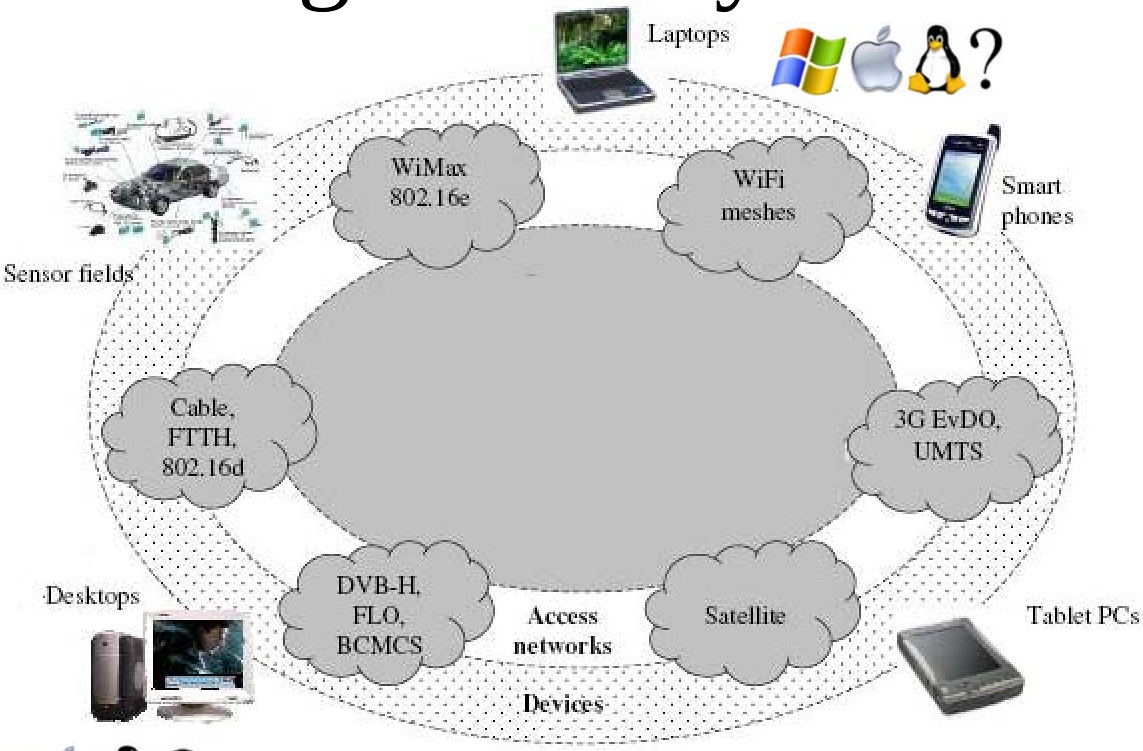
```
•   public final void cook() {
•     beforePlacingOnCookingSurface();
•     placeOnCookingSurface();
•     beforePlacingInCookingDevice();
•     placeInCookingDevice();
•     beforeCooking();
•     for (int i=0; i<getCookingPhases(); i++) {
•       letItCook(getCookTime(i));
•       afterCookingPhase(i);
•     }
•     beforeRemovingFromCookingDevice();
•     removeFromCookingDevice();
•     afterRemovingFromCookingDevice();
•   }
```

Template Method Pattern



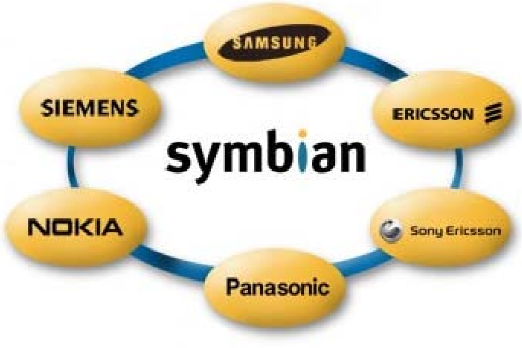
XML and DOM

How can we exchange data between heterogeneous systems?



?

palm



Novell. TOSHIBA EPSON hp IBM NOKIA Connecting People

Message in the Bottle (or: towards the Digital Rosetta Stone)

- Degree of "self-description":

not quite *not bad* *pretty good*

```
^@Some Quotations from the Universal Library^M1
        Famous Quotes^M1.1 By William I^M[2, Sonnet
        XVIII]^MShall I compare thee to a summer's
        day?^MThou art more lovely and more
        temperate.^MRough winds do shake the darling
        buds of May,^MAnd summer's lease hath all too
        short a date.^MSometime too hot the eye of heaven
        shines,^MAnd often is his gold complexion
        dimmed.^MAnd every fair from fair some
        declines,^MBy chance or nature's changing course
        untrimmed.^MBut thy eternal summer shall not
        fade,^MNor lose possession of that fair thou
        owest,^MNor shall Death brag thou wander'st in
        his shade^MWhile in eternal lines to time thou
        growest.^MSo long as men can breathe, or eyes can
        see,^MSo long live this, and this gives life to
        thee.^M1.2 By William II^M[1, p.265]^M\223The
        obvious mathematical breakthrough would be
        development of^Man easy way to factor large
        prime numbers."^MReferences^M[1] W. H. Gates.
        The Road Ahead. Viking Penguin, 1995.^M[2] W.
        Shakespeare. The Sonnets of
        Shakespeare.609.^M^@^@^@^@^@^@^@^@^
        @^@^@^@^@^@^@^@^@^@^@^@^@
```

Two Important Ideas: (1) Markup?

- Information added to a text to make its structure comprehensible

- Pre-computer markup (punctuational and presentational)

Two Important Ideas: (2) declarative

- Names and structure
- Finer level of detail (most human-legible signals are overloaded)
- Independent of presentation (abstract)
- People often call this “semantic”

Message in the Bottle (or: towards the Digital Rosetta Stone)

- Degree of "self-description":

not quite *not bad* *pretty good*

```
^@Some Quotations from the Universal Library^M1 Famous Quotes^M1.1 By William I^M[2, Sonnet XVIII]^MShall I compare thee to a summer's day?^MThou art more lovely and more temperate.^MRough winds do shake the darling buds of May,^MAnd summer's lease hath all too short a date.^MSometime too hot the eye of heaven shines,^MAnd often is his gold complexion dimmed.^MAnd every fair from fair some declines,^MBy chance or nature's changing course untrimmed.^MBut thy eternal summer shall not fade,^MNor lose possession of that fair thou owest,^MNor shall Death brag thou wander'st in his shade^MWhile in eternal lines to time thou growest.^MSo long as men can breathe, or eyes can see,^MSo long live this, and this gives life to thee.^M1.2 By William II^M[1, p.265]^M\223The obvious mathematical breakthrough would be development of^Man easy way to factor large prime numbers."^MReferences^M[1] W. H. Gates. The Road Ahead. Viking Penguin, 1995.^M[2] W. Shakespeare. The Sonnets of Shakespeare.609.^M^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
```

```
\documentclass{article}
 \begin{document}
 \title{Some Quotations from the Universal Library}
...
\section{Famous Quotes}
 \subsection{By William I}
 \textbf{\cite[Sonnet XVIII]{shakespeare-sonnets-1609}}
    \begin{verse}
     Shall I compare thee to a summer's day?\\
     Thou art more lovely and more temperate. \\
     Rough winds do shake the darling buds of May, \\
     And summer's lease hath all too short a date. \\
     Sometime too hot the eye of heaven shines, \\
     And often is his gold complexion dimmed. \\
   ...
     \qquad So long as men can breathe, or eyes can see,\\
     \qquad So long live this, and this gives life to thee.   \\
   \end{verse}
...
 \bibliographystyle{abbrv}
\bibliography{msg}

\end{document}
```

XML: Basic format

1) *Element*: `<tag>content</tag>`

- – basic unit
- – tag name defines what the content is
- – opening and closing tags enclose content

2) *Attribute*: Information about the data

- – Attribute names are usually adjectives
- – Stored as `attribute="value"` pairs:
  - `<tag attribute="value">`
  - `        content`
  - `</tag>`

Rules for well-formed XML

- Elements that contain data must have <start> and </end> tags!

- Empty tags must be closed `<some-tag/>`

- Elements should not overlap

  Bad Nesting:

  `<trunk> <branch> </trunk> </branch>`

- All attribute values must be wrapped in quotes

  `<a href="newpage.html">`

- XML is case sensitive: `<TAG>` and `<Tag>` are treated differently. (Standard: use lower case.)

More XML Rules

- A document begins with:
  - an ***XML Declaration***

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

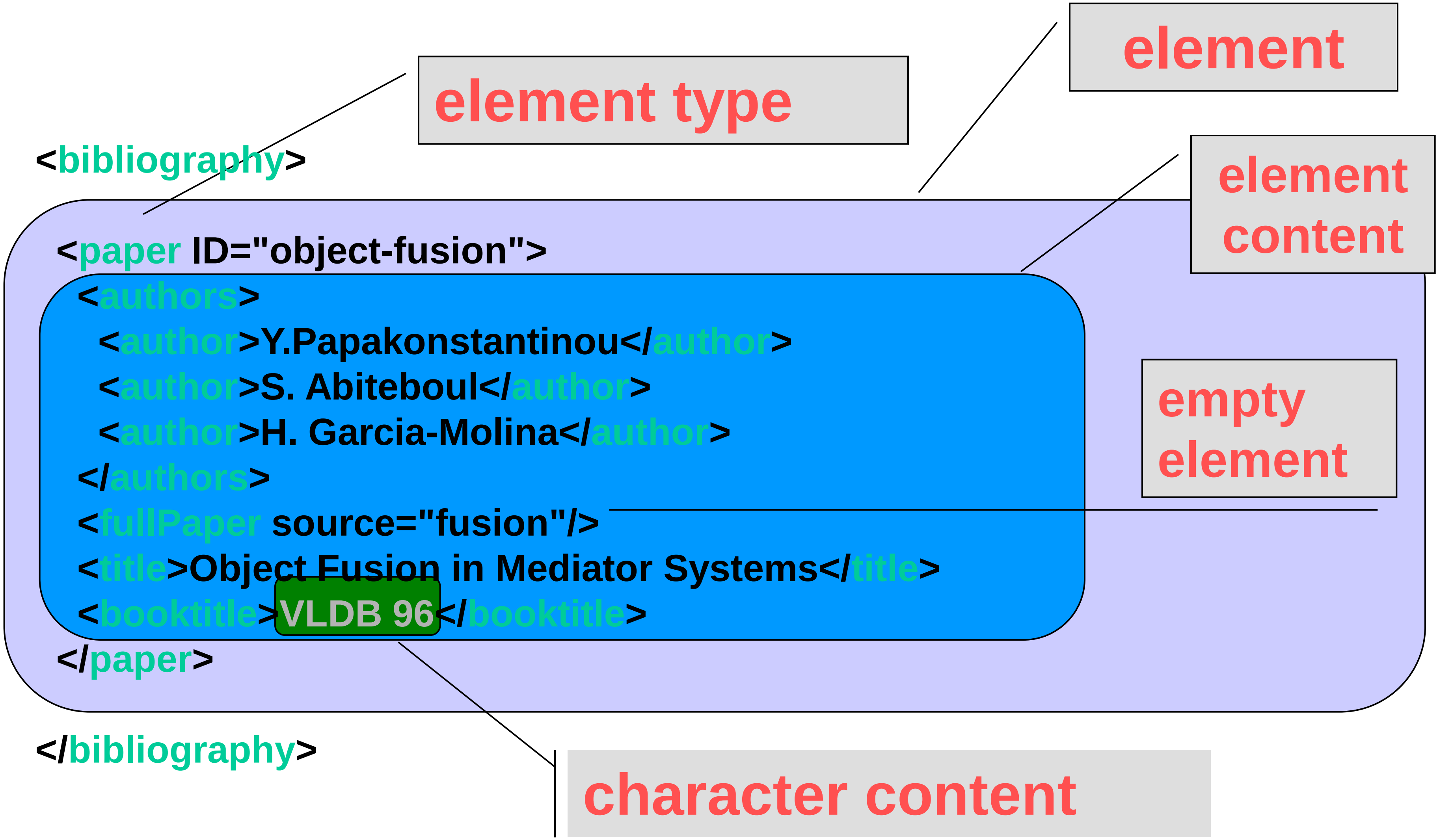
  – and a ***DocType Declaration***:

    ```
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
      1.0 Strict//EN“
      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
      strict.dtd">
    ```

- ***Root element*** immediately follows; encloses entire content of the document.

  ```
      <book>
          everything else
      </book>
  ```

Elements and their Content



```
<bibliography>
  <paper ID="object-fusion">
    <authors>
      <author>Y.Papakonstantinou</author>
      <author>S. Abiteboul</author>
      <author>H. Garcia-Molina</author>
    </authors>
    <fullPaper source="fusion"/>
    <title>Object Fusion in Mediator Systems</title>
    <booktitle>VLDB 96</booktitle>
  </paper>
</bibliography>
```

element type

element

element content

empty element

character content

Element Attributes

Attribute name

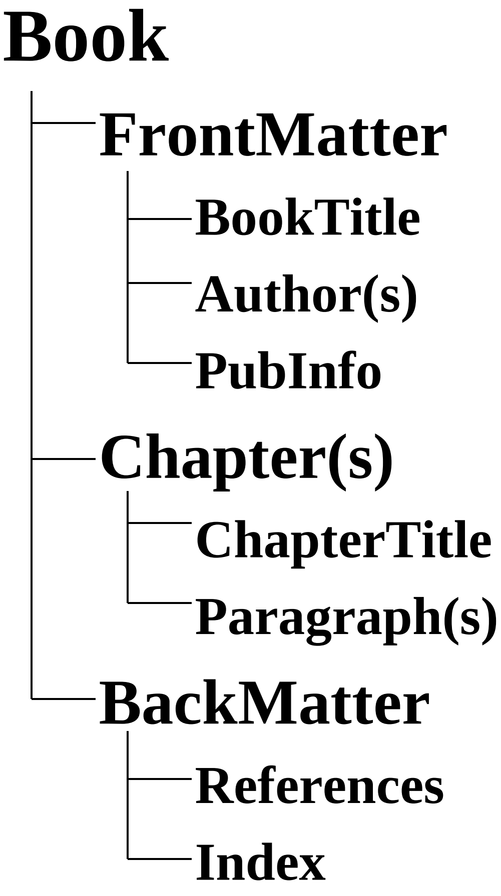
Attribute Value

```
<bibliography>

  <paper pid="object-fusion">
    <authors>
      <author>Y.Papakonstantinou</author>
      <author>S. Abiteboul</author>
      <author>H. Garcia-Molina</author>
    </authors>
    <fullPaper source="fusion"/>
    <title>Object Fusion in Mediator Systems</title>
    <booktitle>VLDB 96</booktitle>
  </paper>

</bibliography>
```

XML Example: content objects in a book



- **Book**
  - **FrontMatter**
    - **BookTitle**
    - **Author(s)**
    - **PubInfo**
  - **Chapter(s)**
    - **ChapterTitle**
    - **Paragraph(s)**
  - **BackMatter**
    - **References**
    - **Index**

A simple XML fragment

```
<Book>
  <FrontMatter>
    <BookTitle>XML Is Easy</BookTitle>
    <Author>Tim Cole</Author>
    <Author>Tom Habing</Author>
    <PubInfo>CDP Press, 2002</PubInfo>
  </FrontMatter>
  <Chapter>
    <ChapterTitle>First Was SGML</ChapterTitle>
    <Paragraph>Once upon a time …</Paragraph>
  </Chapter>
</Book>
```

This is NOT XML, why?

```
<PoemFragment>
  <Stanza>
    <Line><Sentence>It was six men of Indostan</Line>
    <Line>To learning much inclined,</Line>
    <Line>Who went to see the Elephant</Line>
    <Line>(Though all of them were blind),</Line>
    <Line>That each by observation</Line>
    <Line>Might satisfy his mind.</Sentence></Line>
  </Stanza>
</PoemFragment>
```

This is NOT XML, why?

```
<PoemFragment>
 <Stanza>
  <Line><Sentence>It was six men of Indostan</Line>
  <Line>To learning much inclined,</Line>
  <Line>Who went to see the Elephant</Line>
  <Line>(Though all of them were blind),</Line>
  <Line>That each by observation</Line>
  <Line>Might satisfy his mind </Sentence></Line>
 </Stanza>
</PoemFragment>
```

Message in the Bottle (or: towards the Digital Rosetta Stone)

- Degree of "self-description":

not quite — *not bad* — *pretty good*

```
^@Some Quotations from the Universal Library^M1
        Famous Quotes^M1.1 By William I^M[2, Sonnet
        XVIII]^MShall I compare thee to a summer's
        day?^MThou art more lovely and more
        temperate.^MRough winds do shake the darling
        buds of May,^MAnd summer's lease hath all too
        short a date.^MSometime too hot the eye of heaven
        shines,^MAnd often is his gold complexion
        dimmed.^MAnd every fair from fair some
        declines,^MBy chance or nature's changing course
        untrimmed.^MBut thy eternal summer shall not
        fade,^MNor lose possession of that fair thou
        owest,^MNor shall Death brag thou wander'st in
        his shade^MWhile in eternal lines to time thou
        growest.^MSo long as men can breathe, or eyes can
        see,^MSo long live this, and this gives life to
        thee.^M1.2 By William II^M[1, p.265]^M\223The
        obvious mathematical breakthrough would be
        development of^Man easy way to factor large
        prime numbers."^MReferences^M[1] W. H. Gates.
        The Road Ahead. Viking Penguin, 1995.^M[2] W.
        Shakespeare. The Sonnets of
        Shakespeare.609.^M^@^@^@^@^@^@^@^@^
        @^@^@^@^@^@^@^@^@^@^@^@^@
```

```
\documentclass{article}
 \begin{document}
 \title{Some Quotations from the Universal
        Library}
...
\section{Famous Quotes}
 \subsection{By William I}
 \textbf{\cite[Sonnet XVIII]{shakespeare-
        sonnets-1609}}
   \begin{verse}
    Shall I compare thee to a summer's day?\\
    Thou art more lovely and more temperate.
        \\
    Rough winds do shake the darling buds of
        May, \\
    And summer's lease hath all too short a
        date. \\
    Sometime too hot the eye of heaven shines,
        \\
    And often is his gold complexion dimmed. \\
   ...
    \qquad So long as men can breathe, or eyes
        can see,\\
    \qquad So long live this, and this gives life to
        thee.   \\
   \end{verse}
...
 \bibliographystyle{abbrv}
\bibliography{msg}

\end{document}
```

```
<?xml version="1.0"?>
<universal_library>
  <books>
    <book> <title>Some Quotations from the Universal
           Library</title>
      <section> <title>Famous Quotes</title>
        <subsection>  <title>By William I</title>
          <quote bibref="shakespeare-sonnets-1609">
          <title>Sonnet XVIII</title>
          <verse>
            <line>Shall I compare thee to a summer's
          day?</line>
            <line>Thou art more lovely and more temperate.
          </line>
            <line>Rough winds do shake the darling buds of May,
          </line>
          </verse>
   ...
      <subsection>  <title>By William II</title>
        <quote bibref="gates-road-ahead-1995">
          <title>Page 265</title>
          <line>``The obvious mathematical breakthrough would
          be development of an easy way to factor large prime
          numbers.”</line>
        </quote>
      </subsection>
      </section>
</book>
...
</books>
</universal_library>
```

XML Industry Initiatives

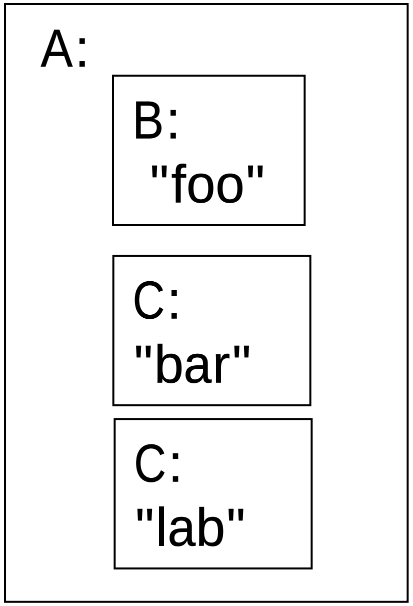
- **Every community is building it's own XML protocols, e.g.:**

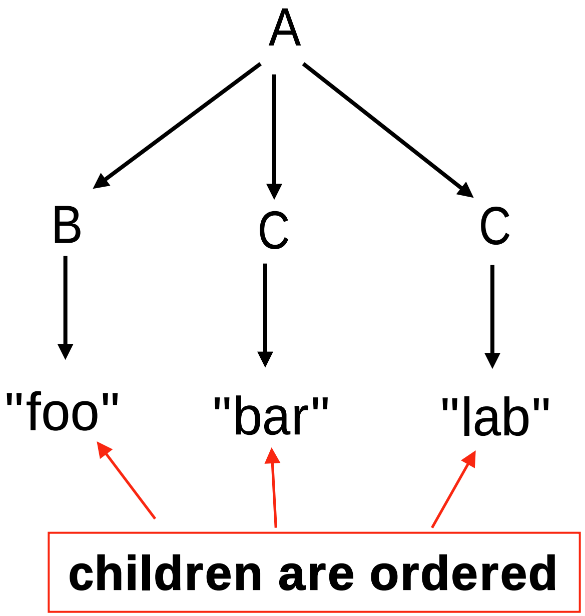
- **Advertising: adXML place an ad onto an ad network or to a single vendor**
- **Literature: Gutenberg convert the world's great literature into XML**
- **Web Servers: apacheXML parsers, XSL, web publishing**
- **Travel: openTravel information for airlines, hotels, and car rental places**
- **News: NewsML creation, transfer and delivery of news**
- **Voice: VoxML markup language for voice applications**
- **Wireless: WAP (Wireless Application Protocol) wireless devices**
- **Weather: OMF Weather Observation Markup Format (simulation)**
- **Geospatial: ANZMETA distributed national directory for land information**
- **Banking: MBA Mortgage Bankers Association of America --> credit report, loan file, underwriting…**
- **Healthcare: HL7 DTDs for prescriptions, policies & procedures, clinical trials**
- **Math: MathML (Mathematical Markup Language)**
- **Surveys: DDI (Data Documentation Initiative) "codebooks" in the social and behavioral sciences**

Http://www.oasis-open.org/cover/xml.html#applications

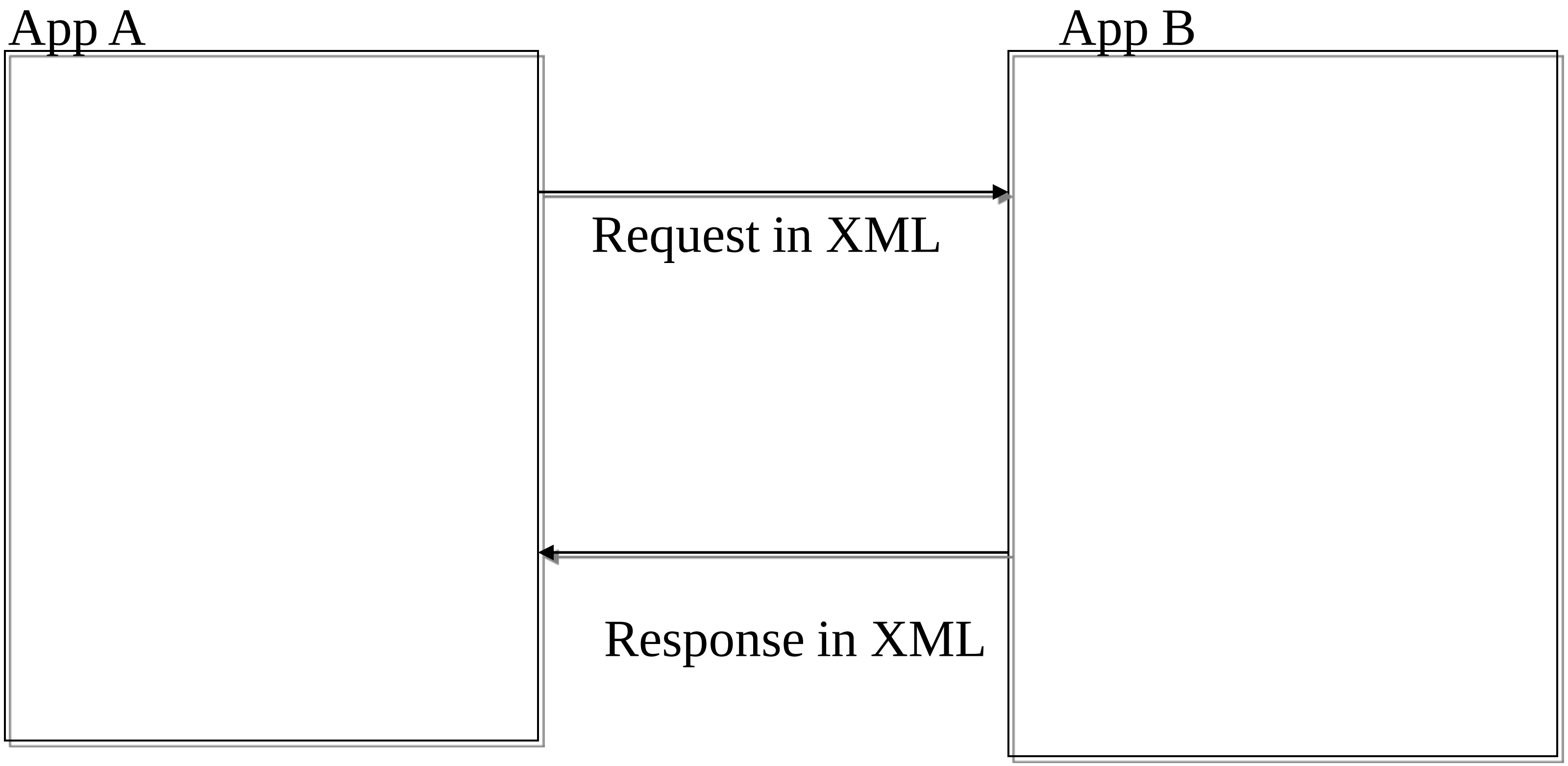
XML -- Instance Model

```
<A>
  <B>foo</B>
  <C>bar</C>
  <C>lab</C>
</A>
```

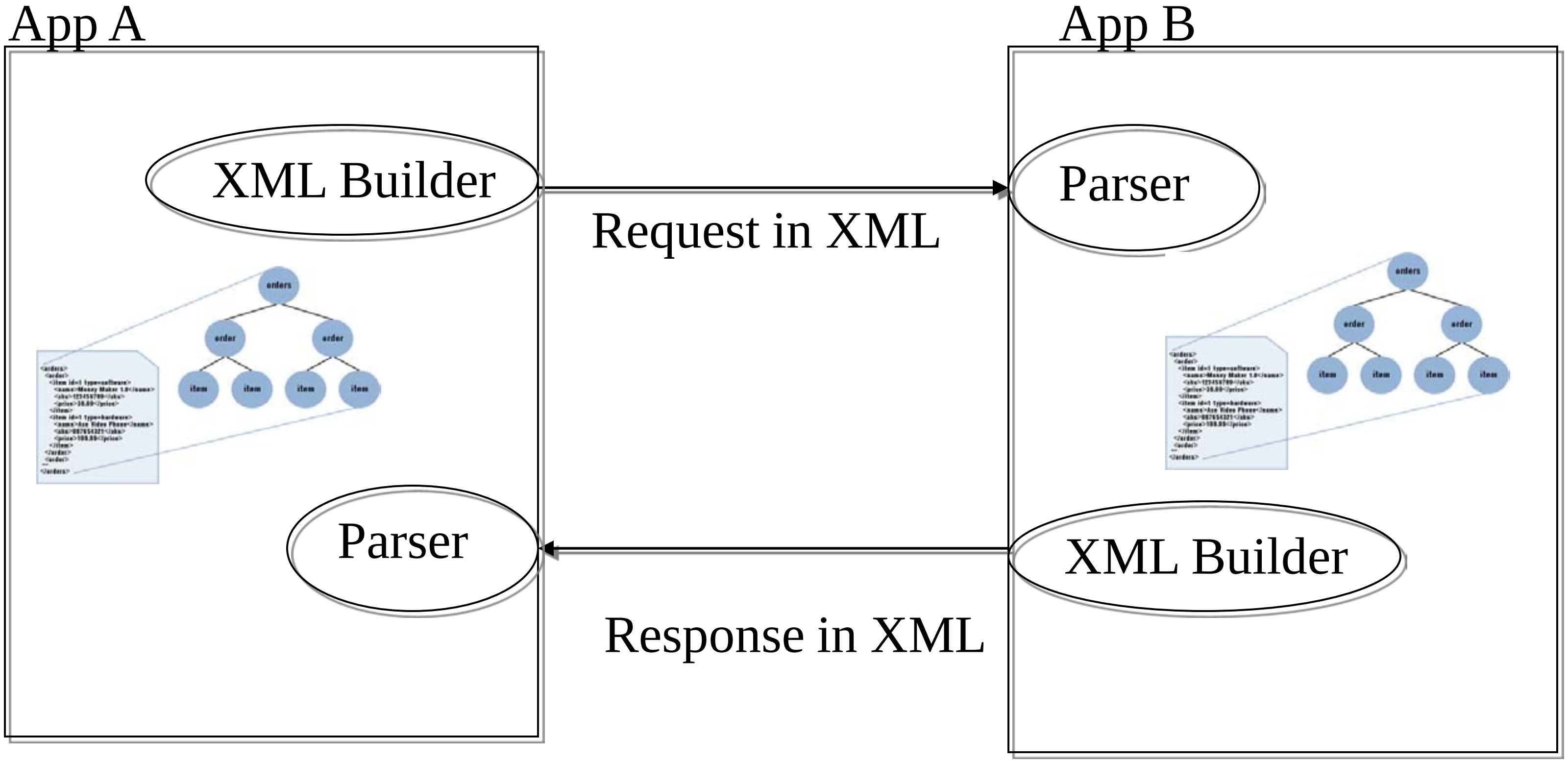




Two Applications Communicating



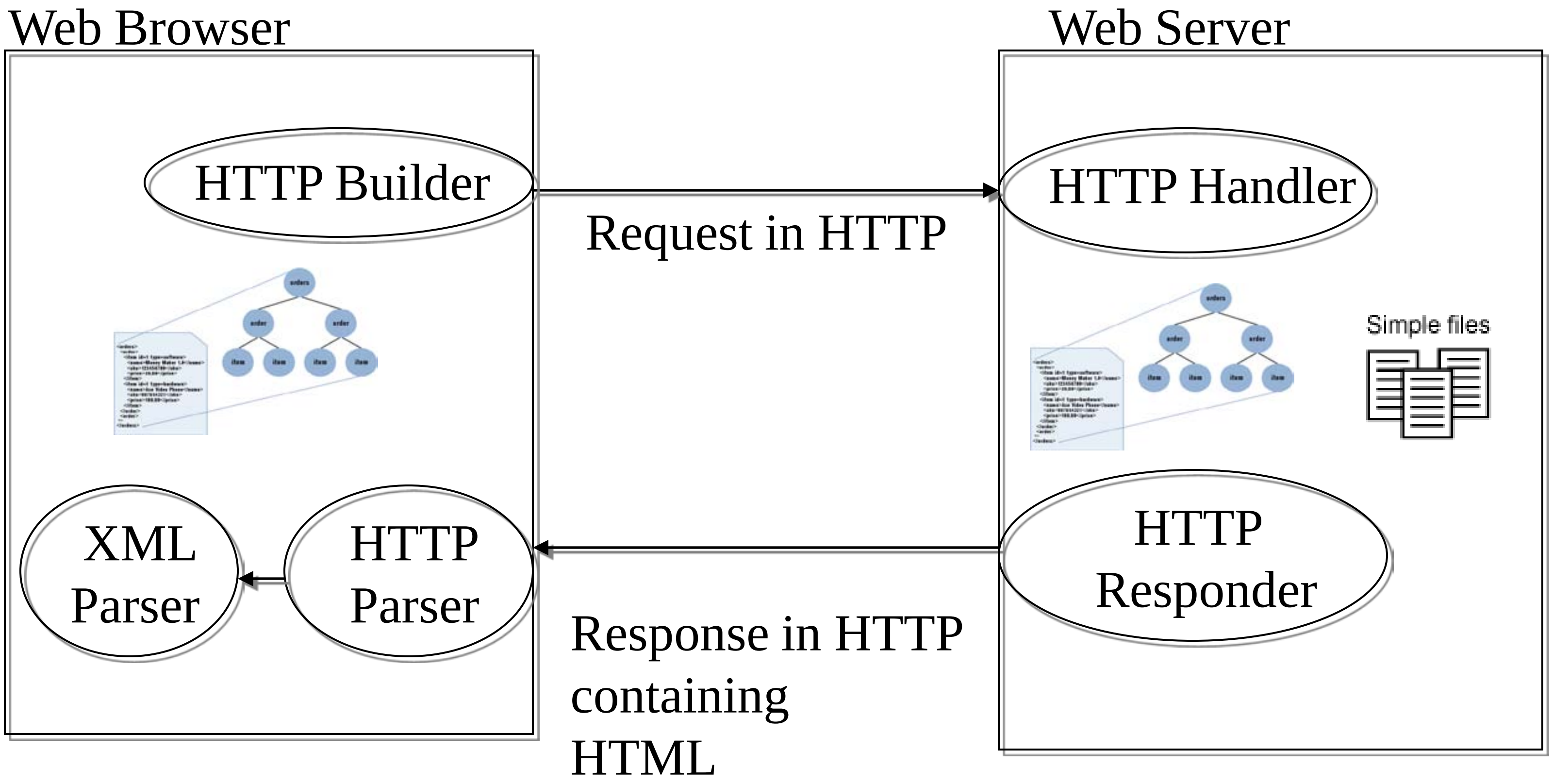
Two Applications Communicating



```
<orders>
  <order>
    <item id=1 type=software>
      <name>Money Maker 1.0</name>
      <sku>123456789</sku>
      <price>39.99</price>
    </item>
    <item id=1 type=hardware>
      <name>Ace Video Phone</name>
      <sku>987654321</sku>
      <price>199.99</price>
    </item>
  </order>
  <order>
  ...
</orders>
```

```
<orders>
  <order>
    <item id=1 type=software>
      <name>Money Maker 1.0</name>
      <sku>123456789</sku>
      <price>39.99</price>
    </item>
    <item id=1 type=hardware>
      <name>Ace Video Phone</name>
      <sku>987654321</sku>
      <price>199.99</price>
    </item>
  </order>
  <order>
  ...
</orders>
```

(X)HTML Case



Web Browser

HTTP Builder

XML Parser

HTTP Parser

Request in HTTP

Response in HTTP containing HTML

Web Server

HTTP Handler

HTTP Responder

Http is a carrier for HTML

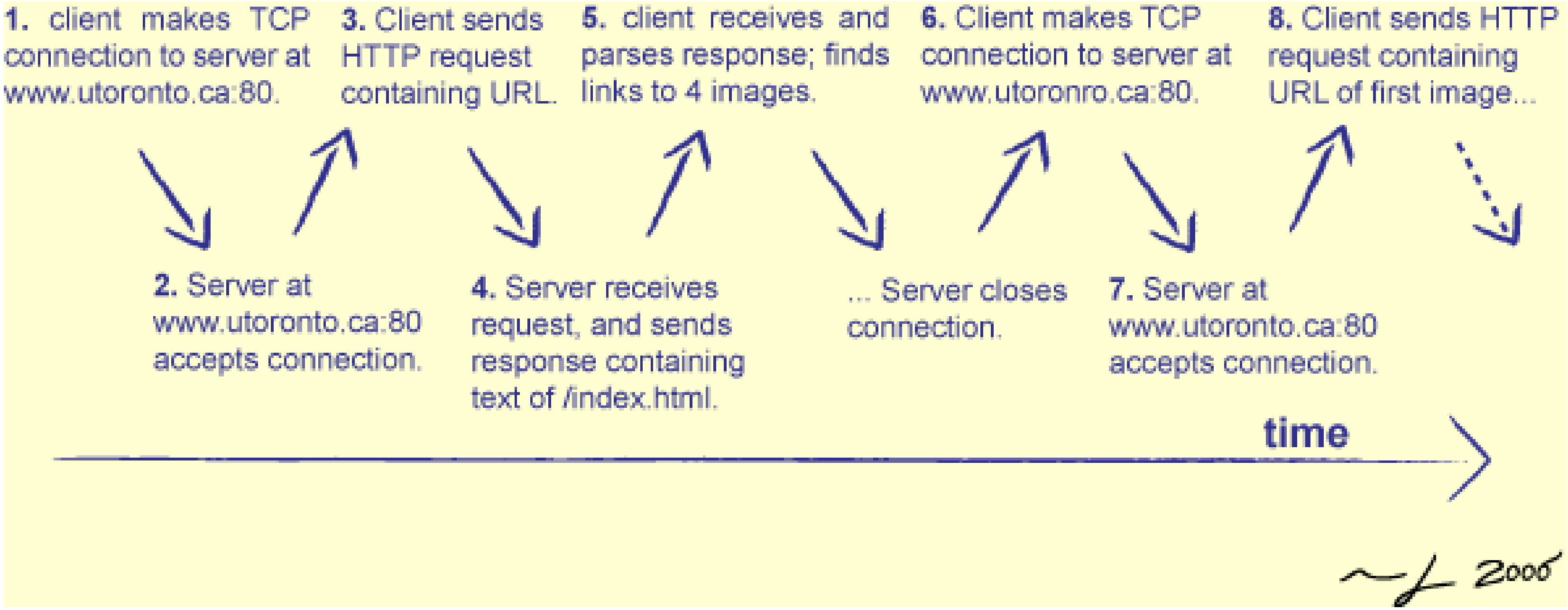
HTTP

- The HyperText Transfer Protocol (HTTP) describes the kinds of messages web servers can receive and send

- HTTP is a *stateless* protocol
  - Each connection is made on its own
  - The web server doesn’t automatically remember anything between connections

- Every transaction is a *request* followed by a *response*

HTTP



1. client makes TCP connection to server at www.utoronto.ca:80.

2. Server at www.utoronto.ca:80 accepts connection.

3. Client sends HTTP request containing URL.

4. Server receives request, and sends response containing text of /index.html.

5. client receives and parses response; finds links to 4 images.

... Server closes connection.

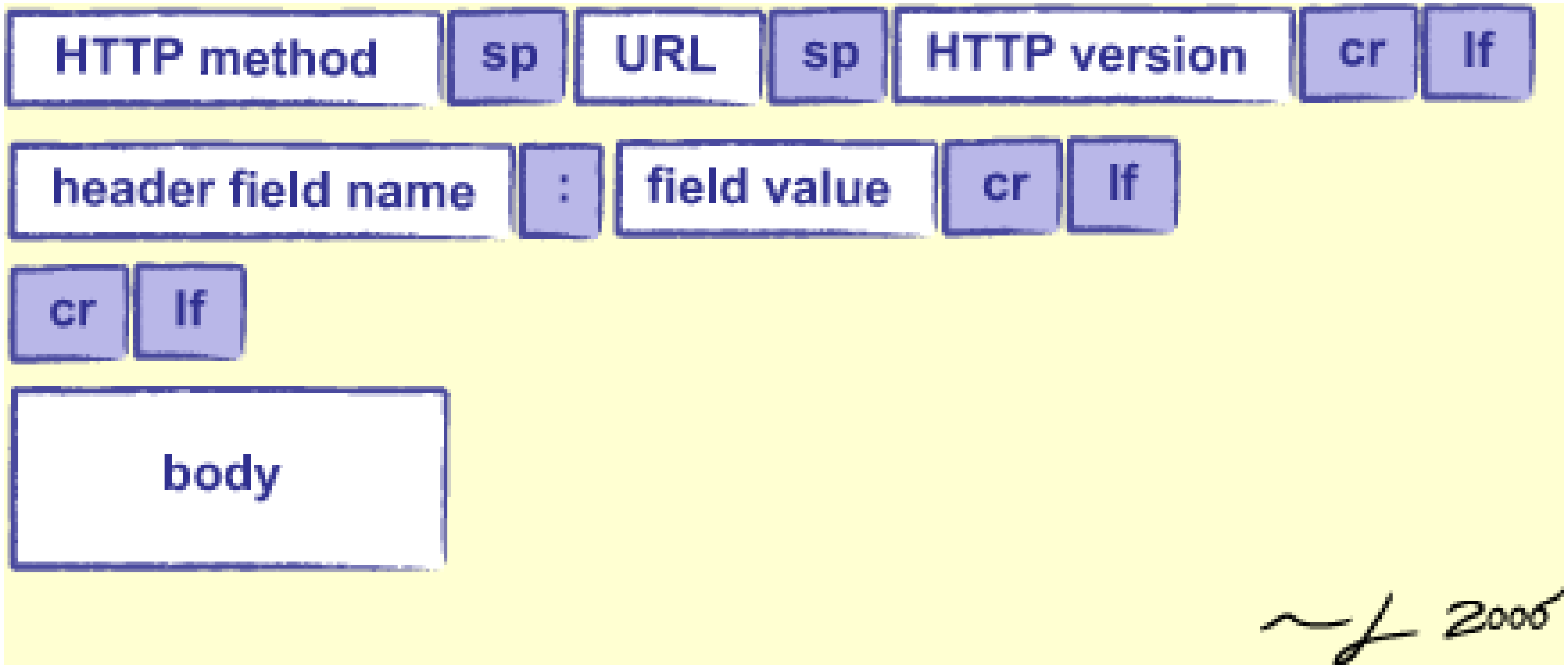
6. Client makes TCP connection to server at www.utoronro.ca:80.

7. Server at www.utoronto.ca:80 accepts connection.

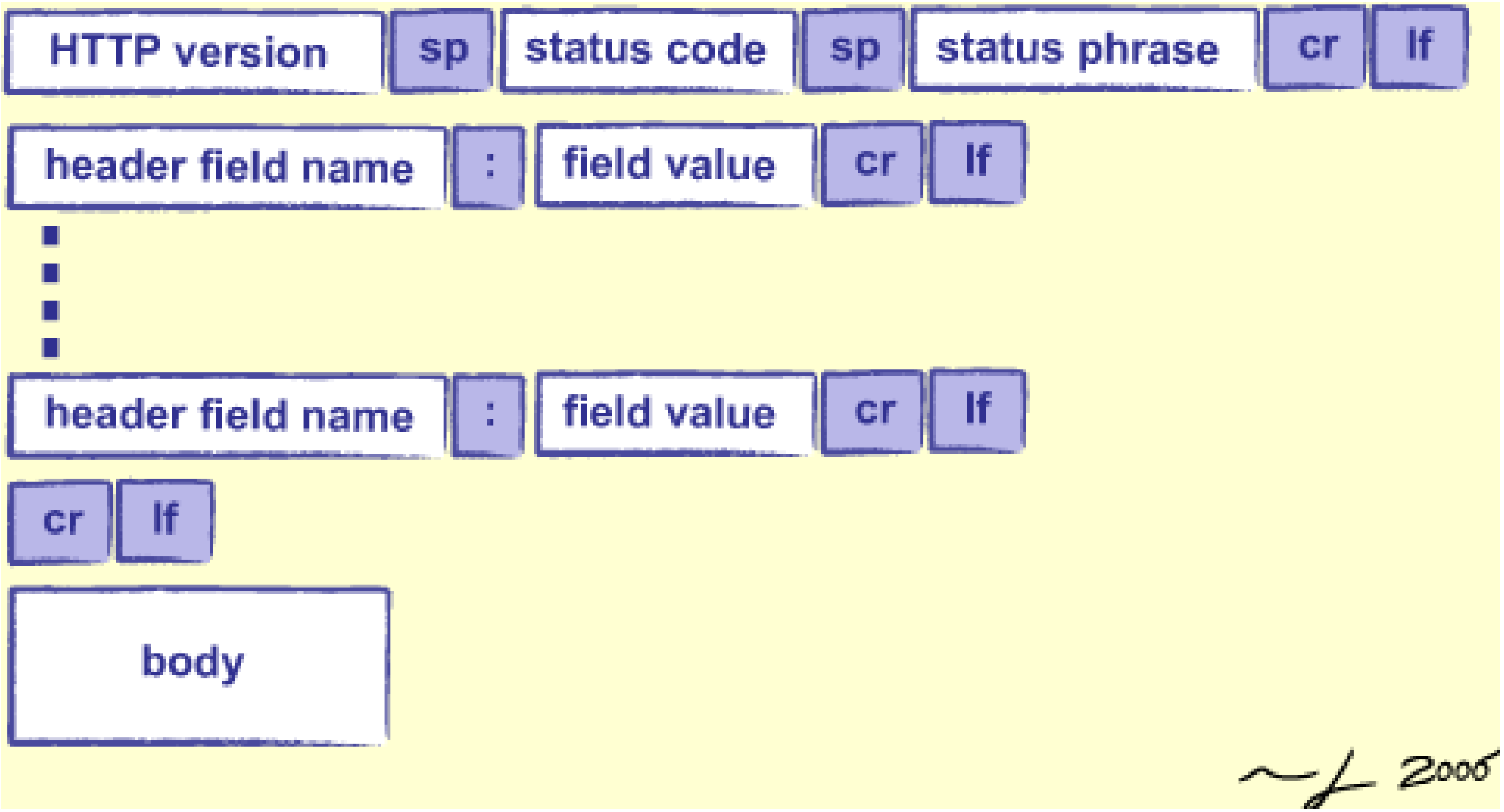
8. Client sends HTTP request containing URL of first image...

2005

HTTP Request



HTTP Response



HTTP version sp status code sp status phrase cr lf

header field name : field value cr lf

header field name : field value cr lf

cr lf

body

2005

HTTP Contents

- HTTP method is usually either:
  - GET (to fetch data)
  - POST (to submit data)
- URL identifies what the client wants
  - Typically a path to a file
  - But the server can interpret however it wants

Headers

- An HTTP header is a key/value pair
  - Accept: text/html
  - Accept-Language: en, fr
  - If-Modified-Since: 16-May-2005
- Unlike a dictionary, a key may appear any number of times
  - So that a request can specify that it’s willing to accept many different kinds of content
- Must be a blank line between the headers and the body!

HTTP Example

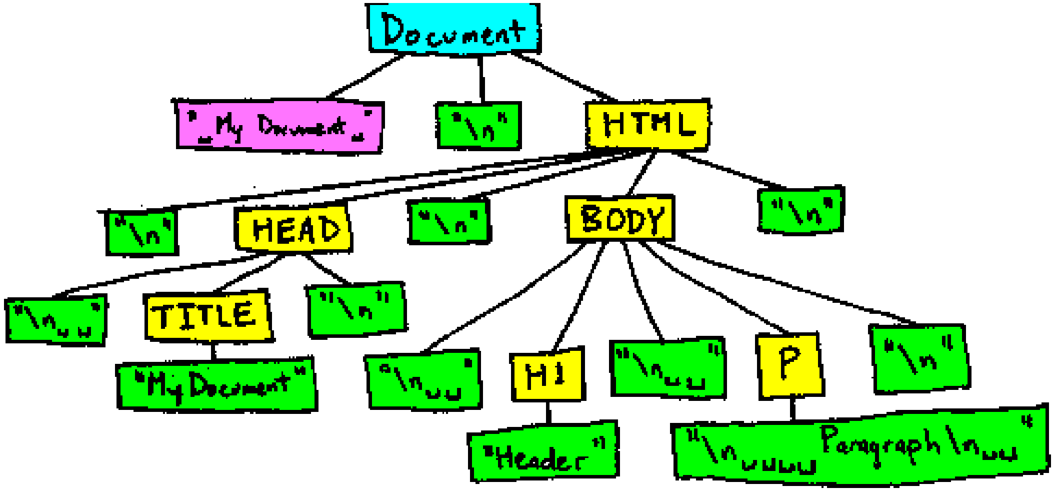
- http://www.rexswain.com/httpview.html

HTML: Historical perspective

- 1989 - Tim Berners-Lee proposed a hypertext system for CERN including HTML and HTTP
- 1993 - Marc Andreessen unleashed the alpha version of Mosaic
- 1993 - (Sept) WWW traffic is 1% of the NFS backbone
- 1994 - more than 200,000 web servers
- 2002 - more than 30,000,000 web servers

HTML Example

```
<!-- My document -->
<html>
  <head>
     <title>My Document</title>
  </head>
  <body>
   <h1>Header</h1>
    <p>
         Paragraph
     </p>
   </body>
</html>
```



http://en.wikipedia.org/wiki/HTTP

Document Object Model (DOM)

- Cross-language API for representing XML documents as trees
  - Easier to manipulate than strings or streams
  - But may require a lot of memory
- Several implementations in Java
  - This course uses `org.jdom`

Tree Structure

- The document:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
  Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
  strict.dtd">
<html>
  <body>
    <h1>Title</h1>
    <p>A <em>word</em></p>
  </body>
</html>
```

DOM rules

- Every document's root is an object of type `Document`
- This has a single child of type `Element`
  - The root element of the document
- Its children may be:
  - Other elements
  - `Text` objects
  - Other things that we won't worry about
- Note: white space is preserved
  - Like the new lines in the previous slide
- But comments are not

Using JDom

```
public static void main(String[] args) {
    try {
      String filename = args[0];
      // Build document tree
      SAXBuilder builder = new SAXBuilder();
      Document doc = builder.build(filename);

      // Show top-level elements (next slide)

    } catch (Exception e) {
      System.err.println(e);
    }
}
```

JDom: Iterate over children

```
// Show top-level elements
Element root = doc.getRootElement();
Iterator ic  = root.getChildren().iterator();

while (ic.hasNext()) {
   Element elt = (Element)ic.next();
   System.out.println(elt.getName());
}
```

Jdom: input and output e.g.

- Input

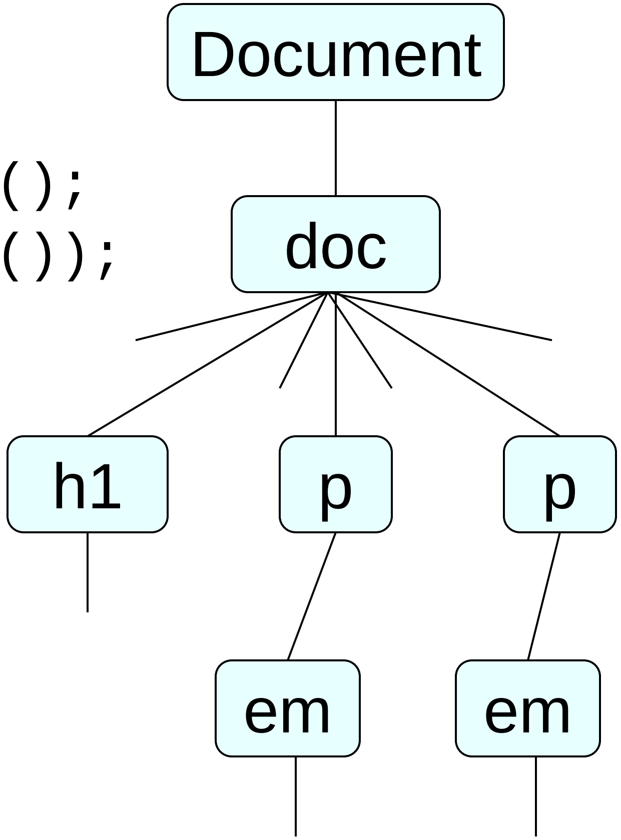
```
<?xml version=“1.0” ?>
<doc>
<h1>First heading</h1>
<p>
  <em> First paragraph</em>
</p>
<p>
  <em>Second paragraph</em>
</p>
</doc>
```

- Output

Jdom: input and output e.g.

```
// Show top-level elements
Element root = doc.getRootElement();
Iterator ic  = root.getChildren().iterator();

while (ic.hasNext()) {
   Element elt = (Element)ic.next();
   System.out.println(elt.getName());
}
```



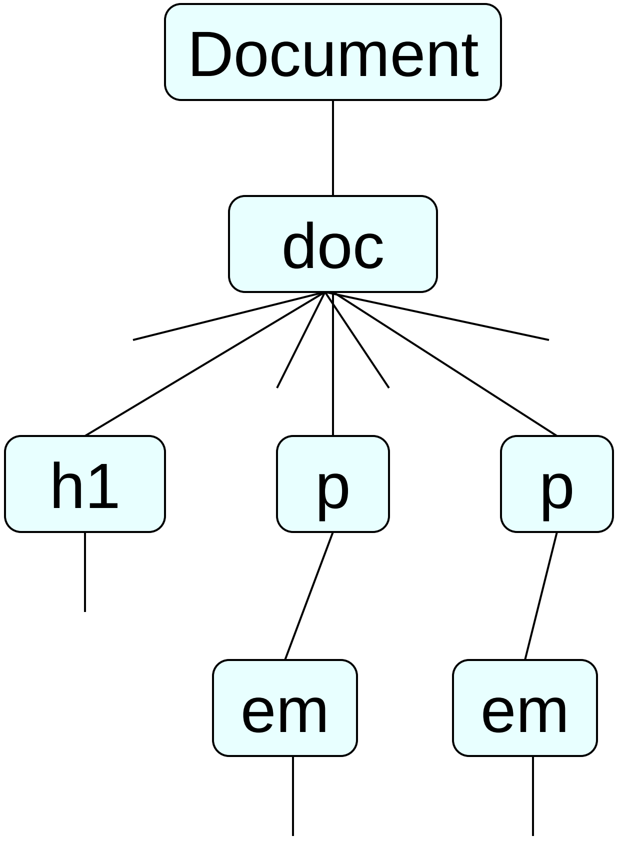
Jdom: input and output e.g.

- Input

```
<?xml version=“1.0” ?>
<doc>
<h1>First heading</h1>
<p>
  <em> First paragraph</em>
</p>
<p>
  <em>Second paragraph</em>
</p>
</doc>
```

- Output

```
h1
p
p
```



Jdom: showing structure recursively

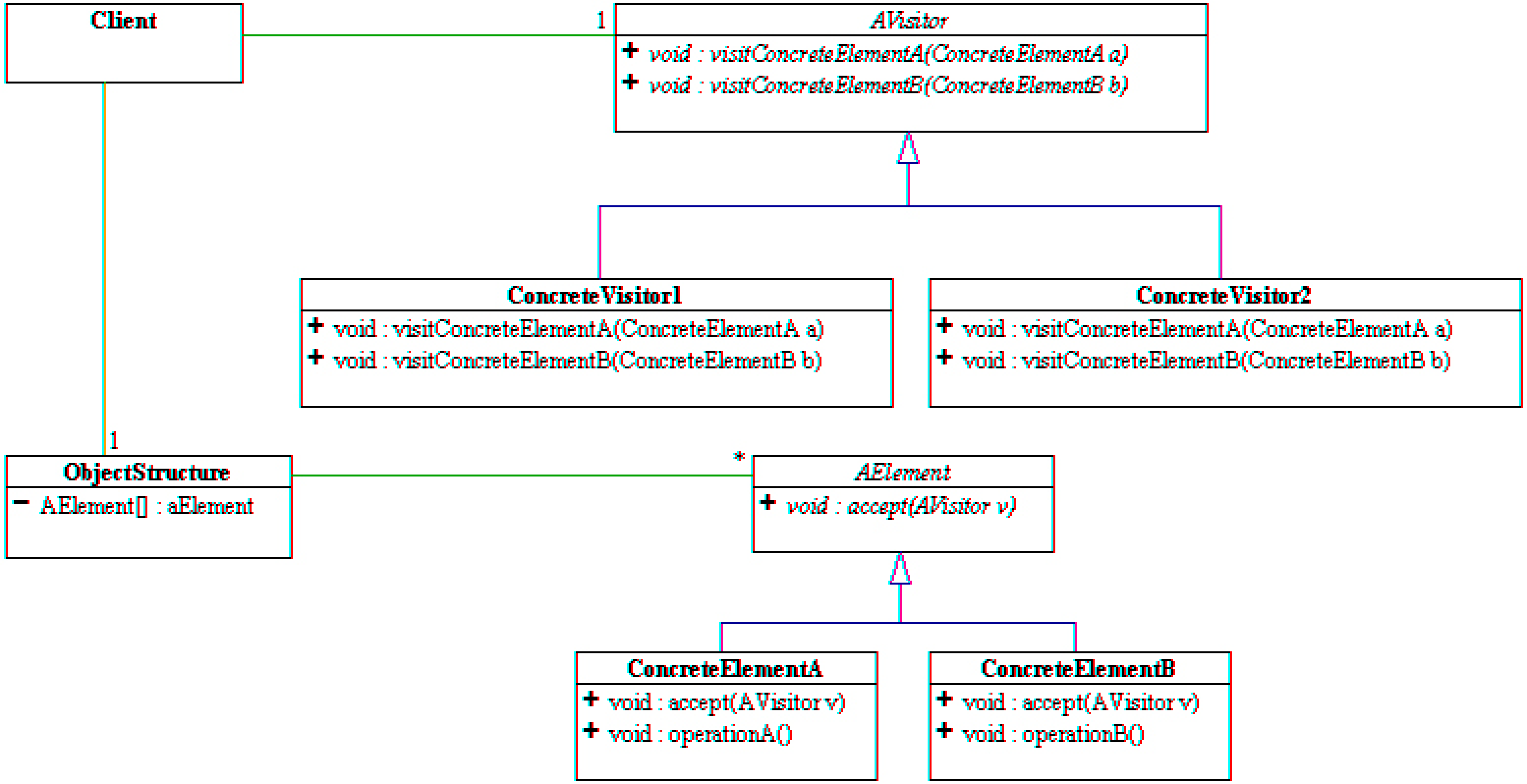
```
  public static void descend(Element current, int depth) {

  for (int i = 0; i < depth; ++i) {
    System.out.print(" ");
  }
  Element elt = (Element) current;
  System.out.println(elt.getName());
  Iterator ic = elt.getChildren().iterator();
  while (ic.hasNext()) {
    descend((Element) ic.next(), depth+1);
  }
}
```

Design Pattern: Visitor

- Often want to operate on a tree recursively
  - Count elements, search for text that matches a pattern, etc.
- Mechanics of traversing is the same every time
- So build a generic visitor that knows how to traverse the tree
  - Give it do-nothing methods that are invoked at specific times during traversal
  - Users derive from this class and override the methods they're interested in

Design Pattern: Visitor


Client
1
AVisitor
+ void : visitConcreteElementA(ConcreteElementA a)
+ void : visitConcreteElementB(ConcreteElementB b)
ConcreteVisitor1
+ void : visitConcreteElementA(ConcreteElementA a)
+ void : visitConcreteElementB(ConcreteElementB b)
ConcreteVisitor2
+ void : visitConcreteElementA(ConcreteElementA a)
+ void : visitConcreteElementB(ConcreteElementB b)
1
ObjectStructure
− AElement[] : aElement
*
AElement
+ void : accept(AVisitor v)
ConcreteElementA
+ void : accept(AVisitor v)
+ void : operationA()
ConcreteElementB
+ void : accept(AVisitor v)
+ void : operationB()

A DOM Visitor

-
  ```
  public abstract class DomVisitor {
    public DomVisitor() {}
    public void visit(Element root) {
    m_depth = 0;
    preRoot(root);
    atElement(root);
    recurse(root);
    postRoot(root);
  }
  ```

…A DOM Visitor

```
protected void preRoot(Element root) {
}

protected void postRoot(Element root) {
}

protected void atElement(Element elt) {
}

protected void atText(Text text) {
}
```

```
  protected void recurse(Element elt) {
    m_depth += 1;
    Iterator ic = elt.getContent().iterator();
    while (ic.hasNext()) {
      Object node = ic.next();
      if (node instanceof Element) {
         Element child = (Element) node;
         atElement(child);
         recurse(child);
      } else if (node instanceof Text) {
         atText((Text) node);
      }
    }
    m_depth -= 1;
  }
}
```

Building an attribute inventory

- Want to find out which attributes appear with which elements in an XML file
- Create a DOM visitor that inspects each element's attributes
- Result is a map in which
  - Keys are element names (e.g. `"h1"`)
  - Values are sets of attribute names (e.g. `"align"`)
- Here we do not record the attribute values
  - Exercise: extend this visitor to inventory them as well

The Inventory Visitor

```
public class Inventory extends DomVisitor {

    public Inventory() {
        m_seen = new HashMap();
    }
    protected void preRoot(Element root) {
        m_seen.clear();
    }
    protected void atElement(Element elt) {
        …
    }
    protected Map m_seen;
}
```

```
protected void atElement(Element elt) {
  String eltName = elt.getName();
  Set seen = (Set) m_seen.get(eltName);
  if (seen == null) {
    seen = new HashSet();
    m_seen.put(eltName, seen);
  }
  Iterator ia = elt.getAttributes().iterator();
  while (ia.hasNext()) {
    String attrName =
           ((Attribute) ia.next()).getName();
    seen.add(attrName);
  }
 }
```

Input and output

```
<doc>
<p align="left“ role="lead">First.</p>
<p align="center">Second </p>
<p align="right" font="em">Third.</p>
</doc>
```

- doc
- p
  - align
  - role
  - font

Trimming the tree

- Can add or remove nodes in DOM tree
  - Be careful about deleting items in a list while iterating over that list
- Pattern: delete or move on
  - When an item is deleted, items above it bump down
  - So either delete *or* increment loop index

```
protected void atElement(Element elt) {
  List content = elt.getContent();
  int i = 0;
  while (i < content.size()) {
    Object node = content.get(i);
    boolean keep = true;
    if (node instanceof Text) {
      Text text = (Text) node;
      if (text.getText().trim().length() == 0) {
         keep = false;
      }
    }
    if (keep) {
      i += 1;
    } else {
      content.remove(i);
    }
  }
}
```

Python

- Like JDOM, Python's DOM library is derived from the W3C standard

- In fact, Python has two DOM libraries
  - xml.minidom doesn't have everything

Example

```
import sys, xml.dom.minidom

def showTree(node, indent=0):
  print '  ' * indent + node.nodeName
  for child in node.childNodes:
      if child.nodeType == child.ELEMENT_NODE:
            showTree(child, indent+1)

for filename in sys.argv[1:]:
  doc = xml.dom.minidom.parse(filename)
  root = doc.documentElement
  showTree(root)
```

Python: urllib

- Python’s urllib hides most of these details

```
import urllib
url ="http://www.thirdbit.com/greeting.html"
instream = urllib.urlopen(url)
lines = instream.readlines()
instream.close()
for line in lines:
      print line
```

Building a Spider

- A *spider* is a program that can explore the web on its own
  - Download a page
  - Use regular expressions to find links
  - Download those pages
  - Repeat

- Oh, and watch out for cycles…

…Building a Spider

```
import sys, urllib, re
url = sys.argv[1]
instream = urllib.urlopen(url)
page = instream.read()
instream.close()
links = re.findall('href=\\"[^\\"]+\\"', page)
temp = set()
for x in links:
   x = x[6:-1] # get rid of href=" and “
   if x.startswith('http://'):
   temp.add(x)
links = list(temp)
links.sort()
for x in links:
    print x
```