XML
References

• http://www.w3.org/TR/xml11: XML 1.1
• www.w3.org/TR/xpath-datamodel: XQuery/XPath data model (XDM)
XML

- A simplified version of SGML
- Designed to substitute HTML
- Became the standard for data exchange and web services
- Some related W3C standards:
  - XPath/XQuery
  - XML Infoset and XDM
  - XSLT
  - DTD, XSD
  - RDF, OWL
  - Many, many others
XML as first conceived

Sample databases included with Access

Microsoft Access provides sample databases.

Northwind Traders database

The Northwind database contains the sales data for a company called Northwind Traders, which imports and exports specialty foods from around the world. By viewing the database objects included in the Northwind database.
<trader ID="T12">
    <name>Wilman Kala</name>
    <address><country>...</country>...</address>
    <orders>
        <order OID="O121">
            <date>1/3/2005</date>
            <item>...</item> <item>...</item>
        </order>
        <order OID="O122">...</order>
    </orders>
</trader>
<trader ID="T13">
    <name>Hanari Cames</name>
    <address><city>...</city>...</address>
    <orders>
        <order OID="T131">
            <date>3/3/2005</date>
            <item>...</item>
        </order>
    </orders>
</trader>
XML syntax

• ‘markup’ and characters
• `<link ref= "./Ag.mdb">Agor&agrave; database</link>`
• `<!-- comment  <>&& -->`
• `<XMLExample>`
  `<![CDATA[<greeting>Hi, world!</greeting>]]>`
`</XMLExample>`
Elements and attributes

- `<link>Agor&agrave; db</link>`
- `<link>`
  *Agor&agrave; `<a>db</a>`*
- `<link ref="./Ag.mdb">Agor&agrave; db</link>`
- `<a/> == <a/></a>`
Entity references

• Entity references:
  – <!DOCTYPE videocollection [
    <!ENTITY R "Romance">
    ...
    <!ENTITY ACT "Action">
  ]>
  – <genre>&R;</genre>

• Predefined ERs:
  – &lt; &gt; &amp; &apos; &quot;
The prologue

• `<?xml version="1.1"?>
  <!DOCTYPE greeting SYSTEM "hello.dtd">
  ...

• `<?xml version="1.1" encoding="UTF-8" ?>
  <!DOCTYPE greeting
     [ <!ELEMENT greeting (#PCDATA)> ]
  >
  ...

• Default version: 1.0
Good formation

• Well formed:
  – Syntax entities are well formed (prologue, elements, attributes, processing instructions, comments)
  – Elements are ‘well nested’
  – No element has two attributes with the same name
  – Every ‘entity reference’ that is used has also been defined
Validity: DTD

• External DTD:
  ```xml
  <!DOCTYPE greeting SYSTEM "hello.dtd">
  <greeting>Hello, world!</greeting>
  ```

• Internal DTD:
  ```xml
  <!DOCTYPE greeting
    [ <!ELEMENT greeting (#PCDATA)> ]>
  <greeting>Hello, world!</greeting>
  ```
DTD: element declaration

- `<!ELEMENT spec (front, body, back?)>`
- `<!ELEMENT div1 (head, (p | list | note)*, div2*)>`
- `<!ELEMENT note (#PCDATA)>`

Means:
- `spec ::= <spec> front body back? </spec>`
- `div1 ::= <div1> head (p | list | note)* div2* </div1>`
- `note ::= <note> string </note>`

- PCDATA: parsed character data
- ‘spec’, ‘front’, etc.: are called ‘types’; ‘(front, body, back?)’ is a content model
Attribute declaration

- `<!ATTLIST list` type `(bullets|ordered|glossary)` "ordered">
- `<!ATTLIST termdef` id ID `#REQUIRED`
  name CDATA `#IMPLIED`>
- `<!ATTLIST form` method CDATA `#FIXED”POST”>`

- T `#Required`: has type T, must be present;
  T `#Implied`: optional;
  T `#Fixed x`: must have “x” as its value;
  T x: “x” default, assigned at validation time
DTDs: main limitations

- No base types apart from PCDATA
- Cannot say that `<address>` inside `<letter>` is different from `<address>` inside `<email>`
- Types cannot be defined by restriction or by extension of other types
- XSD (XML Schema Definition) adds these features, and many others
Semantics of XML

• Is `<a>11</a>` the same as `<a> 11 </a>`?
• Is `<weight>10</weight>` the same as `<weight>010</weight>`?
• `<a>11 12</a>` and `<a>

    11  12

</a>`?
• Order of attributes? Comments?
XML Information Set

• A document has an *Infoset* if it is well formed, and namespaces are correctly used
• *Infoset*: a tree of *information items*:
• 11 kinds of *infoitems*: **Document**, **element**, **attribute**, **PI**, unexpanded ER, **character information**, **comment**, DTD, unparsed entity, notation, **namespace**
• Every *infoitem* has some properties:
  – Element infoitem: namespace name, local name, prefix, children, attributes, namespace attributes, in-scope namespaces, base URI, parent
Post Schema-Validation Infoset

• Validation according XML Schema Definition (XSD) transforms an Infoset in a PSVI:
  – Every infoitem gets an XSD type
  – Missing attributes that have a default value, are initialized with their default value
XDM (XQuery/XPath Data Model)

• XQuery and XPath manipulate XDM values; every value is a sequence of items: atoms or nodes
• XDM is based on PSVI
• A node is essentially a PSVI infoitem, but consecutive CharInfo’s are merged into a unique Text node
• Node identity: a document is a forest-shaped graph \(<N,E>\), and a node is an element of N, with its own identity
  – \(<b>t</b>\) \(\neq\) \(<b>t</b>\)
• Every node has a value and a type
Other details

• 7 types of nodes: document, element, attribute, namespace, PI, comment, text
• Attributes: are NOT children of their parent
• ID – IDREFs: pointers inside a document
• Namespaces to avoid clashes when documents are merged
• Type annotations, validation
A document and its tree (XDM)

• `<a> A
  <b att="zz">B B</b>
  <c/>
</a>`

Document node

Element node node-name=a

Text node content=A

Attribute node node-name=att string-value=zz
Namespaces

• Ref: [http://www.w3.org/TR/xml-names/](http://www.w3.org/TR/xml-names/)
• Amazon defines book with a given structure, B&N define book with a different structure; how to merge them in a unique document?
• Namespaces: every name is a pair URI:local-name
• Every organization uses its own URIs, such as:
  – [http://www.w3.org/1999/xhtml](http://www.w3.org/1999/xhtml)
• A URI is not necessarily a meaningful URL!
The default URI

• Defining a default URI:
• `<xml version="1.0">`?
  <!-- elements are in the HTML namespace, in this case by default -->
<html xmlns='http://www.w3.org/1999/xhtml'>
  <head><title>Frobnostication</title></head>
  <body><p>Moved to <a href='http://frob.example.com'>here</a>.</p></body>
</html>`
Prefixes instead of URIs

• <!-- unprefixed element types are from "books" -->
  <book xmlns="urn:loc.gov:books"
    xmlns:isbn="urn:ISBN:0-395-3631-6">
    <title>Cheaper by the Dozen</title>
    <isbn:number>15684913</isbn:number>
  </book>

• title abbreviates (urn:loc.gov:books, title)
• book abbreviates (urn:loc.gov:books, book)
• isbn:number abbreviates
  (urn:ISBN:0-395-3631-6, number)
Expanded QNames

• QName: book, isbn:number
• An expanded QName is a URI – local name pair, obtained from the QName as follows
  – If there is no prefix, we use the default URI (if defined)
  – If there is a prefix, it is substituted by the associated URI
• Two QNames are equal if, and only if, their expansion is the same
QNames in XDM

• Lexical space (used for input/output): prefix (optional) – local name

• Semantic space (equality and other operations): URI (optional) – local name

• In XDM an expanded QName is a triple: prefix – URI – local name (XDM keeps track of the original prefix)