



XML

eXtensible Markup Language

What are smart data?

- Easy: data that are structured so that software can do useful work with them

Example:

- relational database data
- vector images
- serialized Java objects

So, what are dumb data?

- Data which can only be used for one purpose, because of a lack of structure

Example:

- GIF image with text inside

Can be displayed, but requires OCR software to retrieve the text as text

Smart data on the Web


- Server-side, lots:
 - RDBMSs, SGML, special formats...
- Client-side, hardly any:
 - HTML, PDF, plain text.... **May be?**

This means that the Web consists of millions of pages of information, all of which is very awkward to process automatically

The trouble with this

- Your browser essentially becomes a stupid display client, using HTML instead of other systems
- To work with the data in any way you must talk to software on the server, which performs the actual work
- The HTML in your browser is dump data

The need of exchange

- *business online*  exchange of information between heterogeneous systems
- However, the Web and the internet themselves do not support this in any way
- The alternative: SGML, ASN.1 are large and complex

A critique of HTML

- Extraordinarily flexible, but low on structure
- Fixed tag set (vocabulary)
- No automatic validation
- Unreliable use of the syntax
- Nobody uses the data model

How XML solves this

- Define your own tags (vocabulary)
- Validate against the definition
- Error handling and strict definition of the syntax
- Smaller and simpler than SGML
- Standardized APIs for working with it
- A data model specification is coming

XML background

- A subset of SGML
- Simplifies SGML by:
 - leaving out many syntactical options and variants
 - leaving out some DTD features
 - leaving out some troublesome features
- Standard approved by the W3C

Elements

- A simple and complete XML document:

markup

Start tag

```
<address>
```

Content

```
  <street> 33, Terry  
  Dr.</street>  
  <city> Morrystown </city>
```

End tag

```
</address>
```

Elements

- Attach semantics to a piece of a document
- Have an element type (‘example’, ‘name’) represented by the *markup*.
- Can be nested at any depth
- Can contains:
 - other elements (*sub-elements*)
 - text (*data content*)
 - a combination of them (*mixed content*)

Document Element

- It is the outer element containing all the elements in the document

example:

```
<letter>
```

```
...
```

```
</letter>
```

- It must always exist

Empty Elements

- elements without content
 - They do not have end tags
 - Particular representation of start tags

example:

```
<emptytag/>
```

Attributes

- Used to annotate the element with extra information

- Always attached to start tags:

```
<elem-name attr-name="value" ..>
```

- Elements can have any number of attributes, but all distinct

An XML Document

```
<letter>
  Dear Dr. <receiver http-ref="http://.../GuerriniG.html">
  Guerrini </receiver>, this is my resume.
<resume>
  <name ID="R-212">
    <fname> Dario </fname>   <lname> Bozzali </lname>
  </name>
  <address>
    <street>35, Lake Street</street>
    <city> Morristown </city>
    <email mailto="dario@....com"/>
  </address>
</resume>
</letter>
```


Elements Vs Attributes

Do I use an element or an attribute to store semantic info?

- An **element**, when:
 - I need fast searching process
 - it is visible to all
 - it is relevant for the meaning of the document

- An **attribute**, when:
 - it is a choice
 - it is visible only to the system
 - it is not relevant for the meaning of the document

Other Stuff

- Processing instructions, used mainly for extension purposes (`<?target data?>`)
- Comments (`<!-- ... -->`)
- Character references (`£=£`)
- Entities:
 - named files or pieces of markup
 - can be referred to recursively, inserted at point of reference

Document Types

- Basic idea: we need a type associated with a document, just like objects and values
- A document type is a class of documents with similar structure and semantics

Examples: slide presentations, journal articles, meeting agendas, method calls, etc.

DTDs

- DTDs provide a standardized means for declaratively describing the structure of a document type
- This means:
 - which (sub-)elements an element can contain
 - whether it can contain text or not
 - which attributes it can have
 - some typing and defaulting of attributes

An XML DTD

```
<!ELEMENT letter(sender?,receiver*,(body|
  resume),sign?,#PCDATA)>
. . . .
<!ELEMENT resume(name,address,hobby?)>
<!ELEMENT name(fname,lname)>
<!ELEMENT fname(#PCDATA)>
<!ELEMENT lname(#PCDATA)>
<!ELEMENT address(street,city,(email|http-page))>
. . .
<!ELEMENT email EMPTY>
<!ATTLIST name id ID (#REQUIRED)>
<!ATTLIST receiver http-ref (#IMPLIED)>
<!ATTLIST email mailto CDATA>
. . .
```

Element Type Definition

Attribute List Declaration

Well-formed and Valid Documents

- A document is **well-formed** if it follows the grammar rules provided by W3C.
- A document is **valid** if it conforms to a DTD which specifies the allowed structure of the document

What is XML useful for?

- In a Web context:
 - to simplify Web publishing????
 - to enable advanced client-side applications
 - better searching

Using XML outside the Web

- In general applications:
 - as an exchange format
 - as a cross-language serialization format
 - For a configuration/data files ???

Using XML outside the Web

- In publishing, for structured documents
- E-commerce
- for database exchange and report generation (with XSL)

Some XML applications

- CSD: CORBA Software Description (OMG)
- CDF: Channel Definition Format (MS)
- MathML: Mathematical Markup Lang. (W3c)
- GedML: Genealogical exchange format (M.Kay)
- WIDL: Web IDL (webMethods)
- UXF: UML eXchange Format (J. Suzuki)
- XML-RPC: XML-based RPC (Userland)
- CML: Chemical Markup Language (OMF)

Some more XML applications

- VHG: Virtual HyperGlossary (VHG)
- ICE: Information and Content Exchange (Sun, Adobe et al)
- XSA: XML Software Autoupdate (L.M. Garshol)
- WDDX: Web Distributed Data Exchange (Allaire)
- XHTML: Extensible HTML (W3C)