Introduction to Markup Languages, Terminology, Logical and physical structure of documents

February 17, 2013

1 Markup Languages

1.1 What are Markup Languages?

- Formal (computer) languages that allow to use in addition to the normal text in natural languages also syntactically distinguishable constructs specifying the structure of the text, the meaning of parts, etc., and also allows the text to store its metadata (information about the origin, content, authorship, dating, rights used ...).
- Known markup languages (*markup languages*) are languages for web (HTML, XML, ...), but also others such as typesetting formats of the TeX system, text (documentation/help) formatting tools for the UNIX-like systems nroff, troff. Languages for page description for printing and presentation, namely PostScript or PDF have similar characteristics (text + markup or commands).

1.2 What are Markup Languages?

- Distinguishing characteristics of markup languages in comparison to programming languages is superiority of text (in natural language) over the rest of the content (markup, declarative), so files are often referred to as *documents* .
- The preponderance of the text in natural language may not be true in specific applications.
- For example, XML is used as the format of business exchange (database, table) data, where the marking more than text, and this has the character of a text-recorded data of other types (number, date, logical value).

1.3 The Nature of Markup / 1

There are three main categories according to the nature of markup languages and method of their interpretation:

Presentational markup usually characterizes binary content embedded in text, e.g. classical (older) formats for text editors.

Procedural markup indicating how the processor (processing applications) deals with the text. Usually a sequence of instructions that the sections of the text are to perform. This sequence is consequtively processed while the usual programming constructs (branching, loops, subroutines, variables) are available. Ex.: TeX, PostScript.

1.4 The Nature of Markup / 2

Descriptive markup declaratively defines the document structure and meaning of its parts and does not say exactly what step should be performed while processing – this is usually known by the applications. Ex.: HTML

1.5 Tagging without computers

Around the sixties, the concept of tagging was known only in non-PC contexts:

- The first markup language (informally) were used to processing texts in books and their typesetting.
- Concealers and typographers make the markup on the edge of the paper to indicate what font to select, to make proofreading marks etc.

1.6 Early computer applications of markup / 1

- The first systems for computerized text processing suffered from the fact that their target printing facilities were very different and hence they must have been "programmed".
- The standard GenCode (author William W. Tunnicliffe) was therefore developed, which allowed to mark the general (generic) print output in the text, and a special compiler customized the output for a particular output device.
- The "real father" of markup languages is often considered Charles Goldfarb from IBM, which developed early seventies the language IBM GML (`http://en.wikipedia.org/wiki/IBM\_Generalized\_Markup\_Language`).

1.7 Early computer applications of markup / 2

- On the basis of these two languages was later SGML (Standard Generalized Markup Language (`http://en.wikipedia.org/wiki/Standard\_Generalized\_Markup\_Language`)) was created later, which in fact is not (one) language but a meta-language , ie. standard to define languages.
- A little different way was taken by the TeX markup language of Donald Knuth, 70s and 0s, describing how a typesetting system should place text in a printed document. Frequently, a system of macros LaTeX (Leslie Lamport) is used instead, which adds descriptive / declarative character to TeX (for example, characterized the logical structure of the document).

1.8 Later markup standards – SGML

The first truly widespread and relatively widely applied (in scales, then, of course, incomparable with today's popularity XML ...) was **SGML** .

- It evolved as a modernization of GML, then followed by formalization and subsequent adoption as an ISO standard.
- It is a metalanguage, ie. rules for the design of specific markup languages, SGML instances.

1.9 SGML

- Languages designed according to the rules of SGML are suitable for hand typing – there is less marking than later in XML. However, the existence of a DTD and connection to it to describe the structure of each document were compulsory.
- SGML later, in the late 90 years, became the basis for formulation of XML as a format easier for machine processing, not necessarily requiring to describe the structure of documents for each file.

2 Introduction to XML

2.1 What is XML?

- XML is a standard by the W3C (`http://www.w3.org`) consortium prescribing how to create markup languages.
- It is therefore a metalanguage.
- It is ideologically based on older standards (SGML Structure Generalized Markup Language) – XML can be seen as almost a subset of SGML.
- There are several other standards closely related to XML, such as XML Namespaces, XInclude, XML Base, XML Infoset.
- These standards together with others (XSLT, XSL-FO, XHTML, CSS ...) form a "family" of XML standards.

2.2 Ten principles for the XML standards / 1

from the preamble for XML 1.0 (Third Edition)

1. *XML shall be straightforwardly usable over the Internet.*XML bude přímočaře použitelné na Internetu.
2. *XML shall support a wide variety of applications.*XML bude podporovat širokou škálu aplikací.
3. *XML shall be compatible with SGML.*XML bude kompatibilní se SGML.

2.3 Ten principles for the XML standards / 2

from the preamble for XML 1.0 (Third Edition)

1. *It shall be easy to write programs which process XML documents.*Tvorba programů zpracovávajících XML bude jednoduchá.
2. *The number of optional features in XML is to be kept to the absolute minimum, ideally zero.*Počet volitelných prvků XML standardu bude málo, optimálně nula.
3. *XML documents should be human-legible and reasonably clear.*XML dokumenty by měly být "lidsky" čitelné a rozumně jednoduché.

2.4 Ten principles for the XML standards / 3

from the preamble for XML 1.0 (Third Edition)

1. *The XML design should be prepared quickly.*Návrh XML standardu by měl být rychle hotov.
2. *The design of XML shall be formal and concise.*Návrh XML musí byt formální a správný.
3. *XML documents shall be easy to create.*XML dokumenty bude možné snadno vytvořit.
4. *Terseness in XML literal is of minimal importance.*Úspornost XML značkování není podstatná.

2.5 Characteristics of XML languages / 1

- XML is not a specific markup language, it's a specification determining how the markup languages should look like,
- so it is a "meta-language",
- conceptually a simplification of the SGML standard to facilitate the creation of parsers (analyzers) and applications.
- As each element in an XML document must be closed, the documents need not have a DTD for structure recognition.

2.6 Characteristics of XML languages / 2

- XML builds on a successful implementation of SGML – HTML. It has similar characteristics in terms of the focus on the Internet.
- Serious discussions are held around *binary XML*, which should be equivalent representations of the same model as the "text" XML.

2.7 Current specifications of XML

- original specification (W3C Recommendation) to the W3C XML 1.0: `http://www.w3.org/XML`
- 5th Edition at Extensible Markup Language (XML) 1.0 (Fifth Edition) (`http://www.w3.org/TR/REC-xml`)
- XML 1.1 (Second Edition) (`http://www.w3.org/TR/xml11`) - changes induced by the introduction of *UNICODE 3* , easier *normalization* , the specification of handling procedure for "end of line" characters . XML 1.1 is not bound to specific version of UNICODE, but always on the latest version.

2.8 W3C Activities / 1

XML Coordination Group intermediateworking group, kind of "interface" between different groups of activities and also externally

XML Core Working Group development of major specifications (XML) and closely related ones (*Namespaces in XML, XML Information Set, XInclude*)

2.9 W3C Activities / 2

Efficient XML Interchange Working Group development of standards for effective exchange of XML data with emphasis on portability and platform independence of the individual products (including eg *Binary XML Characterization*)

XML Processing Model Working Group working on the definition of a scripting language for XML, the specification operations over XML data

XML Linking Working Group the now defunct group worked on the development of *XML Linking Language* XLink) and *XML Pointer Language* (XPointer).

2.10 W3C Activities / 3

XML Query Working Group is designing the XML Query Language (*XQuery* and *XPath* - together with XSL Working Group)

XML Schema Working Group Prepares specifications of *W3C XML Schema* to describe the structure, content, or semantics of XML documents.

2.11 What next?

- Neither XML is an "ultimate solution" to all problems of machine data exchange. Development goes on.
- For interactive (rich) web applications (RIA) with intensive server-to-client communications, because of easier interpretability and smaller data, the formats such as JSON (JavaScript Object Notation) are used.

- YAML is used for handwriting structured data.
- These standards will be mentioned during lectures as well. The focus of the course is in XML, derived formats instruments for processing and applications.

3 Inforesources on XML

3.1 Tutorials and papers

- Zvon XML Tutorial: `http://www.zvon.org/xxl/XMLTutorial/General/book\_en.html`
- Tutoriál ke XML na W3 Schools (`http://www.w3schools.com/xml/default.asp`)
- Microsoft XML Tutorial: `http://msdn.microsoft.com/xml/tutorial/`
- 101 XML Tutorials: `http://www.xml101.com/xml/default.asp`
- XML Tutorials at Beginners.co.uk (`http://tutorials.beginners.co.uk`)
- Tutorials at Developerlife.com: `http://developerlife.com`

3.2 Portals on XML

World Wide Web Consortium (W3C) `http://www.w3.org/`

XML Startkabel `http://xml.startkabel.nl` – news, links

Zvon `http://zvon.org` – excellent collection of tutorials, on-line references in many languages, hosted in CZ

XML Cover Pages xml.coverpages.org (`http://xml.coverpages.org`) – daily updated collection of links to articles, standards, software, etc. in XML. Best in this category.

O'Reilly XML.COM `http://xml.com` – articles, tutorials at a high level

IBM DeveloperWorks, sekce XML http://ibm.com/developer/xml (`http://ibm.com/developer/xml/`) – papers, tutorials, software atd. at a high level

3.3 Electronic confs, news, maillists to XML

XML USENET newsgroup `news:comp.text.xml`

XML-DEV `mailto:xml-dev@xml.org` - best known maillist to XML standards (web archive (`http://lists.xml.org/archives/xml-dev/`))

3.4 XML software

- IBM AlphaWorks: `http://www.alphaworks.ibm.com` – alpha-software from IBM to test for free
- Free XML Software (L. M. Garshol): `http://www.garshol.priv.no/download/xmltools/` – probably the best collection of links to non-commercial XML software
- XMLSoftware: `http://xmlsoftware.com` – includes also commercial SW

3.5 More links to XML

- Activities of W3C: `http://www.w3.org/XML/Activity` – specification of standards, conferences, links to SW, reference tools, links
- *What is XML?* na XML.COM: `http://www.xml.com/pub/a/98/10/guide0.html` – one of the intro articles to XML
- XML: XML Quick Syntax Reference Card (`http://www.mulberrytech.com`) – great, simple reference card
- Commented version of XML specification at XML.COM (Annotated XML): `http://www.xml.com/pub/a/axml/axmlintro.html`

4 Resources on XML at FI

4.1 Courses – Fall term

PA165 Enterprise Java – T. Pitner, P. Adámek, M. Kuba, F. Nguyen, P. Kunc, E. Kučírková a D. Tovarňák

PB029 Electronic document preparation – P. Sojka

PV110 Software electronic publications I – P. Sojka

PV173 Seminary of NLP Lab

4.2 Courses – Spring term

IB047 Intro to corpus linguistics and computer lexikography – K. Pala, P. Rychlý

PA105 Technologies of Information Systems II – J. Král

PA154 Corpus Tools – P. Rychlý

4.3 Courses – Spring term

PA156 Dialogue System – I. Kopeček

PV174 Lab of Electronic and Multimedia Apps – P. Sojka

PV030 Textual IS – P. Sojka

PV113 SW electronic publications II – P. Sojka

5 XML document structure

5.1 XML document structure

Fundamental requirement to *all* XML doc: it must be *well-formed*:

1. It contains *prolog (heading)* andexactly one *root element*.Before and after the root element, there can be processing instructions, comments (`Misc`).
2. It meets all the well-formedness constraints given in the specification.
3. Each of the *parsed entities* which is referenced directly or indirectly within the document *is well-formed*.

5.2 XML document structure – further info

Look at tutorial on XML Fundamentals in English (`http://zvon.org/xxl/XMLTutorial/General/book.html`)ToC for XML (`http://zvon.org/comp/m/xml.html`)

5.3 XML document structure

- XML document structure we distinguish between: *physical* and *logical* structure.
- Application programmers are usually interested just on the *logical* structure,
- while for the authors of content, XML editors, processors may also the *physical* structure be important.

5.4 Physical and logical structure

Logical structure A *document* is divided into *elements* (one of them is the *root*), their *attributes*, *text nodes* in elements, *processing instructions*, *notations*, *comments*.

Physical structure One logical doc may be stored in one or more entities; at least in the *document entity*.

5.5 Composition of logical structure

- node (element, attribute, text node, processing instructions, comments)
- element
- attribute
- text node
- processing instructions
- comments

5.6 Prvky logické struktury (česky)

- uzel (element, atribut, textový uzel, instrukce pro zpracování, komentář)
- element
- atribut
- textový uzel
- instrukce pro zpracování
- komentář

5.7 Elements

are objects *delimited by start- and end-tag*, generally:

```
<tagname ...tag_attributes...>
  tag_content
</tagname>
```

```
<body background="yellow">
   <h1>text node -- content of element h1</h1>
   <p>text node -- content of element p</p>
</body>
```

5.8 Elements – empty

If an element is empty (no child elements, neither text content inside), then we write just *empty element tag*, eg.:

```
<tagname tagattribute1 tagattribute2... />
```

```
<hr width='50%'/>
```

Or equivalently (from logical viewpoint):

```
<hr width='50%'></hr>
```

5.9 Attributes

- "Attached to elements", carry "additional info" to elements – eg. its ID, required formatting (style) in case of (X)HTML, or links to other elements
- Conceptually, we could replace all attributes with elements but we keep attributes to maintain readability.
- The *attribute content is NOT further structured* (at least not according to XML standards. An application may see it other way but generally it is not recommended, cf. attributes in relational data model.)
- The *physical order of attributes* in the start tag is NOT important and generally is NOT considered.

5.10 How to write attributes

- An attribute is composed of its *name* and *value*.
- Attributes are inserted in the start tag which may be empty.
- Attribute value is *always* in quotes or doublequotes add separated by a = from the attribute name.
- For *attribute names* the same rules as for element names hold.
- In one element, there can never be *two or more attributes with same name*.
- If namespaces are used, neither two attributes belonging to the same namespace are allowed.

5.11 Attributes – example

```
<hr width='50%'/>

<table border='1'>
   <tr><td>jedna</td><td>dve</td></tr>
   <tr><td>tri</td><td>ctyri</td></tr>
</table>
```

5.12 Text node

- They carry textual information, textual content.
- Eg. in the next sample, the text `ahoj!` is the text node – not the whole element `em`!

  ```
  <em>ahoj!</em>
  ```

5.13 Processing instructions

- *Processing-instructions* are written using `<?target content?>` markup.
- They inform an application about the expected processing or setting.
- They do not carry content.
- `<?xsl-stylesheet href="mystyle.xsl"?>`
- `href` does *NOT* mean an attribute; PIs do not contain attributes!

5.14 Notations

- *Notation* is enclosed in `<!NOTATION name declaration >`
- It is mostly used to describe binary / non-XML entities - eg. images GIF, PNG,...
- It is a *declaration how to process* the binary data.

5.15 Comments

- Similarly to HTML – *comment* is enclosed into `<!--content-->`
- The comment content is `content`, NOT the the whole comment including markup.
- Comments are usually not important for processing but it may depend on application, eg. *Servlet-side Includes (SSI)* use comments.
- Parsers therefore *should be able* to forward comments to the applications.
- *SAX parsers* ignore this in version 1!!! (resp. do so in version SAX2, in Java the package `org.xml.sax.ext`).

5.16 Entity

- Entity is a basic unit of *physical* doc composition. Corresponds to string or file...
- Parsers should process the entities such as the applications do not know about them.

5.17 Document node

We distinguish:

Document node
- parent of the root element
- may contain also PIs, notations, DOCTYPE etc. and

Root element is the core part of an XML doc. In every file, there is just one.

5.18 In more details...

in the next chapter XML family standards.

6 Characters in XML documents

6.1 Characters in XML documents

The specification allows at some places in XML documents (eg. element name, attribute content...) not all characters.It is good to know the meaning of:

- *character sets* (set of characters with respective codes/numbers), tj. attaching the ordinal value to character (eg. Unicode) a
- *character encoding* (from a given set), eg. UTF-8, eg. the ordinal value is encoded into a sequence of bytes

6.2 Unicode and ISO 10646 Standards / 1

Both standards try to resolve the problem: charsets with more than 256 chars.

- Originally 16-bit Unicode: upto 64 K chars, enough for European languages/alphabets, not sufficient for world languages (eg. Chinese).
- 32-bit Unicode: covers "everything".

6.3 Unicode and ISO 10646 Standards / 2

- Nowadays, out of the 32-bit set just the Basic Multilingual Plane (BMP) is used, covering most of the typical languages.
- For names in XML (non-terminal *Qualified Name* – QName) only BMP chars may be used.

Otherwise any Unicode char may be used.

6.4 Unicode encodings

All XML applications (particularly parsers) must be able to process some Unicode encodings. The most common in CZ/SK/EU are:

- 8-bit, traditional: US-ASCII, ISO-8859-2 (ISO Latin 2), Windows-1250 (=Cp1250) – just a subset of Unicode.
- UTF-8: encoding of all chars in Unicode, each char to 1-6 bytes (different), US-ASCII to 1 byte, Czech/Slovak chars to 2 bytes.
- UTF-16: same principle as UTF-8, but 16 bit (2 bytes) word is the basic unit

6.5 Unicode encodings

- UCS-2: direct encoding of Unicode, chars from BMP are directly represented as their ordinal numbers
- UCS-4: dtto, but for whole Unicode at 4 bytes – not efficient, 4 bytes even for US-ASCII, EU-langs...

UTF encodings are the most important for XML, particularly UTF-8 (but parsers must know both).

6.6 Allowed chars

- Any chars from UNICODE upto x10FFFF (except of xFFFE, xFFFF and the range xD800 — xDFFF).
- *names* must be composed of non-whitespace chars: numerals, letters, . (dot) - (comma, minus) _ (underscore) : etc., must start with a letter or _
- Encoding of the UNICODE chars is not important.

6.7 Allowed chars

- Implicitly if not in prolog indicated otherwise, eg.`<?xml version="1.0" encoding="Windows-1250"?>`UTF-8 or UTF-16 is used.
- The distinction between UTF-8 and UTF-16 is done according to the first two bytes of the document entity (ie. file), by so-called byte-order-mark xFFFE.
- If not present, UTF-8 is assumed, thus UTF-8 is the implicit encoding of UNICODE in XML.