XML basics

Corpus Linguistics.

HAP/LAP.

What is XML

- XML, eXtensible Markup Language, standard language for document *markup*
  - Defines a general syntax.
  - Uses simple and readable labels.
  - Labels describe the content of the document, and are inserted along with the text.
  - Syntactic rules markup are defined: how to insert labels within documents, how to define element boundaries, how to write links...

What is XML

- Provides a standard format.
  - Flexible and open:
    - any document can be marked
    - the label set is not predefined: eXtensible
  - Rigid:
    - must comply with a set of basic rules.
    - XML grammar.

Example XML I

```
<?xml version="1.0" encoding="utf-8" standalone="yes" ?>
<?xml-stylesheet href="pertsona.css" type="text/css" ?>
<person>
  <name-surname>
    <name>Pier Paolo</name>
    <surname>Pasolini</surname>
  </name-surname>
  <occupation>cinema director</occupation>
  <occupation>writer</occupation>
</person>
```

XML Characteristics

- Text is marked up. Mark-labels are also text. Everything is text.
- You only need a text editor to view and create XML documents.
    - but there are frameworks for helping in the task.
- Programs have many options to deal with XML documents:
    - Many programming libraries.
    - Document structure accessible.

XML Characteristics

- Not an presentation language. If used correctly, XML describes the document structure and semantics.
- In fact, XML is a markup *metalanguage*
    - Users can define the mark labels, their structure and meaning.
- A markup schema defines the elements and structure for any compliant XML document: XML document types
    - Scheme languages: DTD, XML Schema (W3C), Relax NG, ...
- XML corpora always depend on some schema.

XML Characteristics

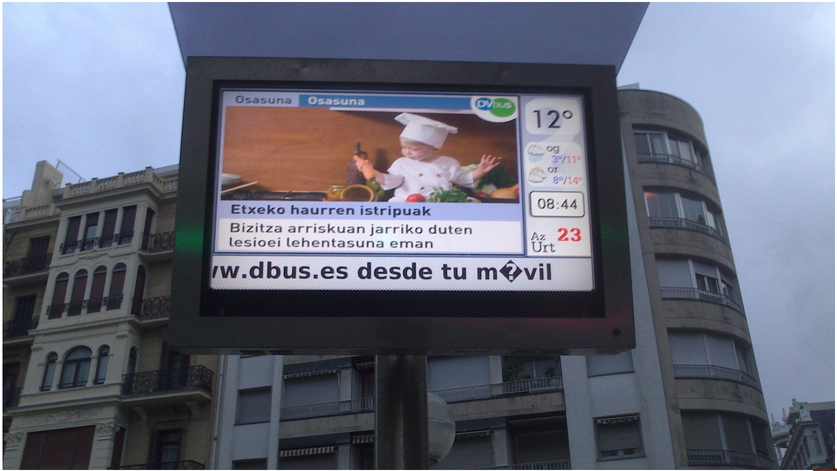
- If the document type is broadly expanded and used by large groups/corporations: XML application
  - For example: XSL, TEI, RSS, MathML,...

Data portability

- Because XML is text, there are no issues regarding different platforms, etc.
  - unlike binary data.
- XML describes the content of the document.
- As a consequence, it is usually used to store, manage, sharing and publication.

Data portability: character encodings

"móvil"



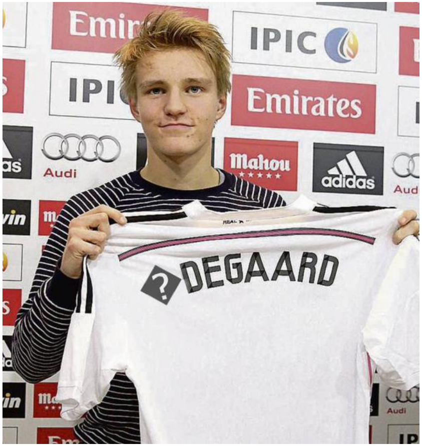
Etxeko haurren istripuak

Bizitza arriskuan jarriko duten lesioei lehentasuna eman

/w.dbus.es desde tu m�vil

Data portability: character encodings

"Martin Ødegaard"



Actually, this is a joke ;-)

Evolution

- XML predecessor: SGML.
- SGML: *Standard Generalized Markup Language*
    - created on the 70s, developed on the 80s.
    - metalanguage for markup, content structure and semantics.
    - but very complex...
- HTML: originally a SGML application
    - tagset for describing web pages
    - focuses on presentation (instead of structure)
    - not meant to be used on anything else.

XML vs. HTML

```
<table border="i">
<tr>
  <th>title</th>
  <th>author</th>
  <th>isbn</th>
</tr>
<tr>
  <td>XML in Action</td>
  <td>William J.</td>
  <td>0-7564-0562-9</td>
</tr>
</table>
```

```
<book>
  <title>XML in Action</title>
  <author>William J.</author>
  <isbn>0-7564-0562-9</isbn>
</book>
```

Evolution

- In 1996 there is an attempt to simplify SGML
  - Motivations:
    - maintain SGML style, but make it easier to use.
    - overcome HTML limitations.
    - better suited for web content (and easier than SGML)
  - Result: XML 1.0 (1998)

Evolution

- New features:
  - Namespaces: different XML applications on same document, no overlap among tags.
  - XSL (*eXtensible Stylesheet Language*): a language to transform XML documents, two main parts:
    - XSLT:*XSL Transformations*: general language to transform XML documents.
    - XL-FO: *XSL Formatting Objects*: general language for rendering XML documents.

Evolution

- Link and reference content and elements
  - XLink (*eXtensible Linking Language*): language to describe links (intra- and inter-document).
  - XPath: general specification for referring document parts.
  - Xpointer: similar to XPath, but including inter-document elements.

Evolution

- Programming issues
  - DOM (*Document Object Model*): API to manage XML documents. Requires leading full document in memory.
  - SAX (*Simple API for XML*): event-centric API. Events are generated along with the document parsing.

Evolution

- Schema languages:
    - DTD (*Document Type Definition*): original schema language, derived from SGML.
    - XML Schema (W3c)
    - Relax NG
    - ...

Evolution

- Many technologies related to XML
  - XML native databases.
  - XML XQuery: for querying XML databases. Equivalent to SQL in relational databases.
  - XInclude: Create XML documents by including pieces from several places.
  - XML *Encryption*: XML standard to encrypt documents.
  - ...

Documents and Elements

- XML documents follow a strict syntax (so that parsing is efficient).
- The basic component is the *element*
- Elements consist of
  - start and ending *tag*
  - Textual content

Element example

```
<name>            <!-- start tag -->
  Pier Paolo      <!-- textual content -->
</name>           <!-- end tag -->
```

Syntax of elements

- Start tag: <name>
- End tag: </name>
  - end-tags are required.
- Element's tags describe the element.
- XML is case sensitive.
- There can be empty elements: <postag/>
- Elements can be nested, but overlapping is not allowed.

Valid	Invalid
`<book>` `<title>XML in Action</title>` `</book>`	`<book>` `<title>XML in Action` `</book>` `</title>`

Valid:

```
<book>
  <title>XML in Action</title>
</book>
```

Invalid:

```
<book>
  <title>XML in Action
</book>
</title>
```

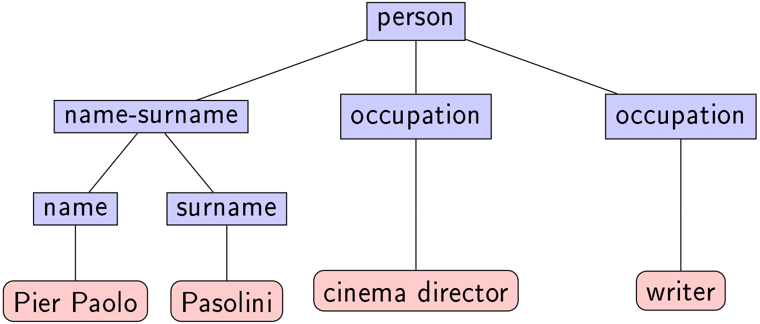
XML trees

- XML documents are parsed as a tree.
- The following XML document:

```
<?xml version="1.0" encoding="utf-8" standalone="yes" ?>
<person>
  <name-surname>
    <name>Pier Paolo</name>
    <surname>Pasolini</surname>
  </name-surname>
  <occupation>cinema director</occupation>
  <occupation>writer</occupation>
</person>
```

is equivalent to this tree:

Syntax of elements



Another example

```
<?xml version="1.0" encoding="utf-8" ?>
<library>
  <book code="XML1">
    <title>XML in action</title>
    <author>William J. Pardi</author>
    <isbn>0-7356-0562-9</isbn>
  </book>
  <journal code="CL1">
    <title>Computational Linguistics</title>
    <publisher>ACL</publisher>
    <publisher>MIT Press</publisher>
    <issn>0891-2017</issn>
  </journal>
</library>
```

Mixed elements

- Sometimes raw text is mixed with elements:

```
<biography>
<name-surname><name>Bernardo</name>
<surname>Bertolucci</surname></name-surname>, <occupation>cinema
director</occupation>, started his career as a pupil of <name-surname>
<name>Pier Paolo</name> <surname>Pasolini</surname></name-surname>
, from which he got his style...
</biography>
```

- *Data-centric XML*: communication between machines.
- *Document-centric XML*: narrative markup.

Attributes

- Elements may contain attributes.
- Attributes are (key, value) pairs, described at the start tag.
- Values must be enclosed in quotes (simple or double)

```
<person id="G1900000" source="wikipedia">
  <name-surname>
    <name>Pier Paolo</name>
    <surname>Pasolini</surname>
  </name-surname>
  <born>1922</born>
  <died>1975</died>
</person>
```

Attributes

```
<person source="wikipedia">
  <born>1922</born>
  <name-surname>
    <name>Pier Paolo</name>
    <surname>Pasolini</surname>
  </name-surname>
  <occupation type="primary">cinema director</occupation>
  <occupation type="secondary">writer</occupation>
</person>
```

- When to use children / attributes ?
  - One guideline:
    - Use attributes for meta-information.
    - Use children for information.
  - Take into account:
    - one element can not have more than one attribute with the same name.
    - attribute values have no structure.

XML names

- the XML specification put the same restrictions to a name, regardless of its use (element name, attribute name, etc).
- In summary
    - Must begin with letter or underscore (_).
    - After initial character following are allowed digits:
        - hyphen (-)
        - underscore(_)
        - colon(:) - legal but should be used except for namespaces.
        - period(.)
    - NO other characters are allowed like #, , $, %

Entity references

- We want to write the character “<” in a XML document.
  - but “<” is always interpreted as element start.
- Solution: use entity references
  - Use “&lt;” for encoding “<”
- Same problem if we want to write the character “&” ...

Entity references

`&lt;` (`&#60`)	<
`&amp;` (`&#38;`)	&
`&gt;`	>
`&quot;`	"
`&apos;`	'

- `&lt;` and `&amp;` are mandatory. The rest can be infered by the context.
- Besides the predefined entities, users can define many more (DTD).

CDATA sections

- We can insert a whole raw section without interpreting the characters.

```
<description> attribute <code> svg </code> can be used, e.g.:
  <![CDATA[
    <svg xmlns="http://www.w3.org/2000/svg"
         width="12cm" height="10cm">
      <ellipse rx="110" ry="150" />
    </svg>
  ]]>
</description>
```

Comments

- You can put comments into XML documents.
- Comments are no elements.
- Where to put them: any place, but outside tags
- Rules:
  - between strings `<!--` and `-->`
  - double hyphens (`--`) disallowed inside comments.
  - last character of comment can not be a hyphen (`-`)

```
<person source="wikipedia">
  <!-- this is a comment -->
  <name-surname>
    <name>Pier Paolo</name>      <!-- another comment -->
    <surname>Pasolini</surname>
  </name-surname>
</person>
```

Processing instructions

- Allow documents to contain instructions for applications.
- Not elements.
- Where to put them: any place, but outside tags
- Rules:
  - between strings <? and ?>
  - may have “pseudo-attributes”

```
<?robots index="yes" follow="no" ?>
```

Processing instructions

```
<?xml-stylesheet type="text/xsl" href="person.xsl"?>
<person>
  Pier Paolo Passolini
</person>
```

- This example associates the XML document with a stylesheet.
- The PI comes before any element.
- It tells the browser to execute script `person.xsl` and render the output.

XML declaration

```
<?xml version="1.0" encoding="utf-8" standalone="yes" ?>
```

- Not customary.
- If present, has to be the first line of the file.
- Looks like a PI, but it is not.

XML declaration

```
<?xml version="1.0" encoding="utf-8" standalone="yes" ?>
```

- `encoding`:
  - optional attribute (default value `utf-8`).
  - describes current encoding of document.
- `standalone`:
  - Optional attribute (default `no`).
  - if value is “`no`”, an external DTD is required to deal with the document.
  - if “`yes`”, document alone is enough.

Well formed document

- All XML documents have to be *well formed*.
  - If a document is not well formed, it is no XML.
- Any parser tests well-formedness before any further proceesing.

Well formed: rules

There are many rules. The most important are:

1. All start tags have a corresponding end tag.
2. Elements can be nested, but they do not overlap.
3. There is one (and only one) root element.
4. Attribute values between quotes.
5. No duplicated attributes in elements.
6. Syntax in comments (no double hyphens, ...).
7. "<" and "&" characters are forbidden (use entities).
8. XML declaration, if present, has to be the first line.
9. ...

Well formed: how to test

- Load XML document on web browser (Firefox, Chrome, ...)
- Use XML parser/analyzer (for example, `xmllint`).
- Use XML environment (emacs, `<oXygen/>`, etc).