


ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zürich

Module 2

XML Basics

(XML, Namespaces, Infoset, Schema)



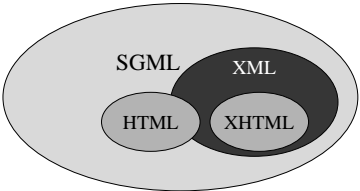
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zürich

Agenda

- Slides cover two lectures!
- XML Core
- Namespaces
- DTD: Structural Constraints for XML
- XML Schema: Complex constraints, data types

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zürich

SGML vs. HTML vs. XML



ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zürich

XML Data Example

```
<book year="1967">
  <title>The politics of experience
</title>
  <author>
    <firstname>Ronald</firstname>
    <lastname>Laing</lastname>
  </author>
</book>
```

- Syntax, no abstract model
- Documents, elements and attributes
- Tree-based, nested, hierarchically organized structure

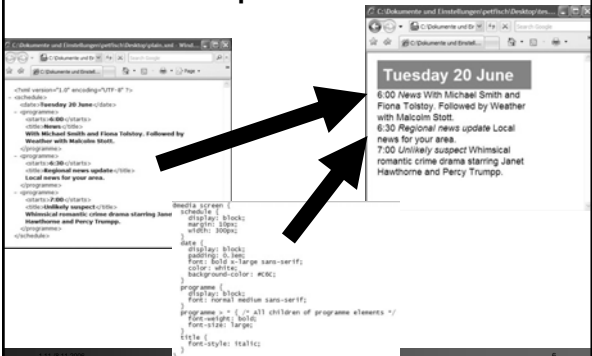
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zürich

Layout of XML Documents

- XML only defines a document structure
- Interpretation for layout is depending on the application
- Using XML (not XHTML) in a browser shows plain tags
- Format (but no interpretation) using CSS
- CSS = Cascading Style Sheets, format instructions for HTML
- Try this in Firefox or IE!

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zürich

XML+CSS Example



ETH
 Eidgenössische Technische Hochschule Zürich
 Swiss Federal Institute of Technology Zürich

XML Components

- Elements
- Comments
- Processing Instructions
- Attributes
- Namespace Declarations
- Text
- Document = Prolog + Root Element + Text
 + Comments + Processing Instructions

7

ETH
 Eidgenössische Technische Hochschule Zürich
 Swiss Federal Institute of Technology Zürich

Elements

- Enclosed in Tags
 - Begin Tag: e.g., <bibliography>
 - End Tag: e.g., </bibliography>
 - Element without content: e.g., <bibliography />
- Elements can be nested
 <bib> <book> Wilde Wutz </book> </bib>
- Subelements can implement multisets
 <bib> <book> ... </book> <book> ... </book> </bib>
- Documents must be well-formed
 <a> is forbidden!
 <a> is forbidden!

8

ETH
 Eidgenössische Technische Hochschule Zürich
 Swiss Federal Institute of Technology Zürich

Attributes

- Attributes are associated to Elements
 <book price = „55“ currency = „USD“ >
 <title> ... </title>
 <author> ... </author>
 </book>
- Element that only has an attribute
 <person name = „Wutz“ age = „33“/>
- What is the difference between a nested element and an attribute? Are attributes useful?
- Attribute names must be unique! (No Multisets)
 <person name = „Wilde“ name = „Wutz“/> is illegal!

9

ETH
 Eidgenössische Technische Hochschule Zürich
 Swiss Federal Institute of Technology Zürich

ID and IDREF

- XML is hierarchical
- Difficult to express N:M relations
- Need references
- What is the target of a reference?
 - No inherent *visible* identity of XML elements
 - Use **id** attribute
- How to place a reference
 - Use **idref** attribute

10

ETH
 Eidgenössische Technische Hochschule Zürich
 Swiss Federal Institute of Technology Zürich

ID and IDREF example

```

<family>
<person id = „27“ >
  <name> Lilly Potter </name>
  <child>
    <person id = „35“ >
      <name> Harry Potter </name>
      <parents idref = „27 42“ />
    </person>
  </child>
</person>
<person id = „42“ >
  <name> James Potter </name>
  <child idref = „35“ />
</person>
</family>
  
```

11

ETH
 Eidgenössische Technische Hochschule Zürich
 Swiss Federal Institute of Technology Zürich

Comments, PIs, Prolog

- Comment: Syntax as in HTML
 <!-- This is just a comment -->
- Processing Instructions
 - Contain no data - interpretation by processor
 - Syntax: <?pause 10 secs ?>
 - Pause is „Target“; 10secs is „Content“
 - XML is a reserved target for prolog
- Prolog
 <?xml version=„1.0“ encoding=„UTF-8“ standalone=„yes“ ?>
 <?xml-stylesheet href=„demo.css“ type=„text/css“ ?>
 - Standalone defines whether there is a DTD
 - Encoding is usually Unicode.

12

ETH
 Eidgenössische Technische Hochschule Zürich
 Swiss Federal Institute of Technology Zürich

Text and Mixed Content

- Text appears in attributes, elements, ...
- Mixed Content:


```
<a> text1 <b/> text2 </a>
```

 - For „documents“ very frequent (e.g. Paragraph with Enumerations)
 - Bad for „data“ processing

Peter Fritzsche, Department of Informatics, ETH Zürich

ETH
 Eidgenössische Technische Hochschule Zürich
 Swiss Federal Institute of Technology Zürich

Whitespace

- Whitespace = Text of Space, Tabs and Returns
- Special Attribute `xml:space` to control use
- Human-readable XML (with Whitespace)


```
<book xml:space="preserve" >
  <title>Die wilde Wutz</title>
  <author>D.A.K.</author>
</book>
```
- (Efficient) machine-readable XML (no WS)


```
<book xml:space="default" ><title>Die wilde
Wutz</title><author>D.A.K.</author></book>
```
- Performance improvement: ca. Factor 2.

Peter Fritzsche, Department of Informatics, ETH Zürich

ETH
 Eidgenössische Technische Hochschule Zürich
 Swiss Federal Institute of Technology Zürich

XML vs. Relational

- XML
 - Allows any (inhomogeneous) trees
 - Typing is optional, 19 base types
 - XML Data are ordered!
- Relational
 - Trees of depth two only
 - All sub-trees have the same structure
 - Typing is strict, ca. 7 base types
 - Data is not ordered
- Theory: XML and Relational are „equivalent“

Peter Fritzsche, Department of Informatics, ETH Zürich

ETH
 Eidgenössische Technische Hochschule Zürich
 Swiss Federal Institute of Technology Zürich

XML vs. Relational

name	phone
John	3634
Sue	6343
Dick	6363

```
<row name = "John" phone = "3634"/>
<row name = "Sue" phone = "6343"/>
<row name = "Dick" phone = "6363"/>
```

Peter Fritzsche, Department of Informatics, ETH Zürich

ETH
 Eidgenössische Technische Hochschule Zürich
 Swiss Federal Institute of Technology Zürich

Namespaces

- Integration of Data from diverse data sources
Integration of different XML Dialects
- Data sources use the same Name, but
 - Different meaning (Teekesselchen)
 - Or different structure
- Idea: Qualified Names for Attributes, Elements
`qname ::= [prefix:] localname`
 - Prefix is optional, refers to a URI
 - Prefix and Localname are separated by „:“
- „<http://w3.org/TR/1999/REC-xml-names>“

Peter Fritzsche, Department of Informatics, ETH Zürich

ETH
 Eidgenössische Technische Hochschule Zürich
 Swiss Federal Institute of Technology Zürich

Example: Namespaces

- DQ1 defines dish for tableware
 - Diameter, Volume, Decor, ...
- DQ2 defines dish for satellite
 - Diameter, Frequency
- How many „dishes“ are there?
- Better ask for:
 - „How many dishes are there?“
or
 - „How many dishes are there?“

Peter Fritzsche, Department of Informatics, ETH Zürich

Example: Namespaces

```
<gs:dish xmlns:gs = „http://tableware.com“ >
  <gs:dm gs:unit = „cm“>20</gs:dm>
  <gs:vol gs:unit = „l“>5</gs:vol>
  <gs:decor>Gelsenkirchener Barock</gs:decor>
</gs:dish>
<sat:dish xmlns:sat = „http://fernsehen.com“ >
  <sat:dm>200</sat:dm>
  <sat:freq>20-2000MHz</sat:freq>
</sat:dish>
```

Several Namespaces

```
<gs:dish xmlns:gs = „http://tableware.com“
          xmlns:uom = „http://units.com“>
  <gs:dm uom:unit = „cm“>20</gs:dm>
  <gs:decor>Gelsenkirchener Barock</gs:decor>
  <gs:vol uom:unit = „l“>5</gs:vol>
  <comment>I am unqualified</comment>
</gs:schüssel>
```

Default Namespaces

tableware.com if not specified otherwise.

```
<schüssel xmlns = „http://tableware.com“
          xmlns:uom = „http://units.com“>
  <dm uom:unit = „cm“>20</dm>
  <decor>Gelsenkirchener Barock</decor>
  <vol uom:unit = „l“>5</vol>
  <kommentar>I am qualified</kommentar>
</schüssel>
```

Namespace Notes

- Namespace definitions look like Attributes
 - Identified by „xmlns:prefix“ or „xmlns“
- URIs used to identify a namespace uniquely
 - But nobody interprets them
 - www.dangling.com is okay
 - Aliases irrelevant
- Scope is Element and Subelement, no Attrs


```
<n:e xmlns n=„http://n.com“ n:a = „illegal“>
  <n:e n:a = „legal“/>
</n:e>
```

Documents Format Descriptions

- XML is a meta-format
- Easy to start with, use your own tags
- Only restriction: XML needs to be well-formed (close your tags!)
- At some point, this is too much freedom
 - Use same syntax for different documents
 - Exchange data with other parties
- Need to restrict the amount of freedom
- Document Description Methods
- DTD, XML Schema

Document Type Definition (DTD)

- Optional for a Document
- Defines Structure of Documents
 - Nesting of Elements, possible and mandatory
 - Attributes of Elements
 - Possible Values of Attributes
 - How often subelement occur in nested elements
- Four kinds of Declarations
 - Notation, Entity, Element Type, Attribute List
- Based on context-free grammars / BNF

ETH
 Eidgenössische Technische Hochschule Zürich
 Swiss Federal Institute of Technology Zürich

DTD Example

```
<!ELEMENT book (title, (author+ | editor), publisher?)>
<!ATTLIST book
  year CDATA #REQUIRED
  isbn ID #REQUIRED
  price CDATA #IMPLIED
  curr CDATA #FIXED "EUR"
  index IDREFS "" >
<!ELEMENT author (firstname, lastname)>
<!ELEMENT firstname (#PCDATA)>
<!ELEMENT lastname (#PCDATA)>
<!ELEMENT title (#PCDATA)>
```

25

ETH
 Eidgenössische Technische Hochschule Zürich
 Swiss Federal Institute of Technology Zürich

Element Type Declaration

- Structure: `<!ELEMENT name inhalt>`
- Beispiel


```
<!ELEMENT book (title, (author+ | editor), publisher?)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT author EMPTY>
<!ELEMENT publisher ANY>
```
- Valid document according to this DTD


```
<book >
  <title>Die wilde Wutz</title>
  <author/> <author/></author>
  <publisher><anything>...</anything></publisher>
</book>
```

26

ETH
 Eidgenössische Technische Hochschule Zürich
 Swiss Federal Institute of Technology Zürich

Element Type Declarations

- Element Types are composed of:
 - Subelements (identified by Name)
 - Attribute lists (identified by Name)
 - Selection of Subelemente (choice)
 - PCDATA
- Quantifier for Subelements and Choice
 - „+“ for at least 1
 - „*“ for 0 or more
 - „?“ for 0 or 1
 - Default: exactly 1
- EMPTY and ANY are special predefined Types

27

ETH
 Eidgenössische Technische Hochschule Zürich
 Swiss Federal Institute of Technology Zürich

Attribute Lists

- Structure: `<!ATTLIST ElementName definition>`
- ```
<!ATTLIST book
 isbn ID #REQUIRED
 price CDATA #IMPLIED
 curr CDATA #FIXED „EUR“
 index IDREFS „“ >
```
- Valid and Not-valid Books
 

```
<book isbn=„abc“ curr=„EUR“/> !! no price
<book isbn=„abc“ price=„30“/> !! Curr, index default
<book index=„DE“ isbn=„abc“ curr=„EUR“/>
<book/> !! Missing isbn Attribute
<book isbn=„abc“ curr=„USD“/> !! wrong currency
```

28

ETH  
 Eidgenössische Technische Hochschule Zürich  
 Swiss Federal Institute of Technology Zürich

## Attribute Types

- CDATA: normal text
- NMTOKEN: restricted Text
- NMTOKENS: Enumeration, „ “ as separator
- ID
  - Value is unique within document
  - Element has at most one attribute of this type
  - No default values allowed
- IDREF, IDREFS
  - References to other elements within the document
  - IDREFS: Enumeration, „ “ as separator
- ENTITY, ENTITIES, NOTATION
  - See Entity and Notation Declarations in DTD

29

ETH  
 Eidgenössische Technische Hochschule Zürich  
 Swiss Federal Institute of Technology Zürich

## Attribute Defaults

- #REQUIRED**
  - Document must specify a value for attribute
- #IMPLIED**
  - Attribute is optional, there is no default
- value**
  - Default value, if no other value specified
- #FIXED value**
  - Default value, if no other value specified
  - If value specified, it must be the fixed value

30

## Entity and Notation

- ENTITY Declarations for Text replacement
  - <!ENTITY muster „Die wilde Wutz hat Husten“>
  - Serve as Macros
  - General vs. Parametrized Entities
  - Internal vs. External Entities
  - parsed vs. unparsed Entities
- NOTATION Declaration
  - <!NOTATION gif SYSTEM „gifviewer.exe“>
  - e.g. command for special Data types

## DTDs and Namespaces

- Limited support:
- Names in Declarations of DTDs can be qualified names.

## Declarations of DTDs in XML document

- None (well-formed Documents)
- In Document:
 

```
<!DOCTYPE name [definitions] >
```
- External, specified by URI:
 

```
<!DOCTYPE name SYSTEM „demo.dtd“>
```
- External, Name and optional URI:
 

```
<!DOCTYPE name PUBLIC „Demo“>
```

```
<!DOCTYPE name PUBLIC „Demo“ „demo.dtd“>
```
- In Document + External:
 

```
<!DOCTYPE name1 SYSTEM „demo.dtd“
```

```
 [local definitions] >
```

## XML Schema - Motivation

- DTDs provide structural description of XML – so why another, more complex standard?
- How do you compute 3 + 5?
- DTDs only knows „strings“ (PCDATA or CDATA or ...).
- What about Integer, Dates, boolean?
- What about sub-typing? Inheritance?
- What about composite keys?
- These are typical database or programming language issues, not document issues

## Overview XML Schema

- ComplexTypes and SimpleTypes
  - ComplexType correspond to Records/Objects/Structs
  - „string“ is an example of a SimpleType
- Built-in and user-defined Types
  - ComplexTypes are always user-defined
- Elements and Attributes have complexTypes
- Type of Root element of a document is global
- (almost) downward compatible with DTDs
- Schemas are XML Documents (Syntax)
- Namespaces etc. are part of XML Schemas

## Example Schema

```
<?xml version=„1.0“ ?>
```

```
<xsd:schema
```

```
 xmlns:xsd=„http://w3.org/2001/XMLSchema“>
```

```
 <xsd:element name=„book“ type=„BookType“/>
```

```
 <xsd:complexType name=„BookType“>
```

```
 <xsd:sequence>
```

```
 <xsd:element name=„title“ type=„xsd:string“/>
```

```
 <xsd:element name=„author“ type=„PersonType“
```

```
 minOccurs=„1“ maxOccurs=„unbounded“/>
```

```
 <xsd:element name=„publisher“
```

```
 type=„xsd:anyType“/>
```

```
 </xsd:sequence>
```

```
 </xsd:complexType>
```

```
</xsd:schema>
```

ETH  
 Eidgenössische Technische Hochschule Zürich  
 Swiss Federal Institute of Technology Zürich

### Example Schema

```
<?xml version="1.0" ?>
<xsd:schema
 xmlns:xsd="http://w3.org/2001/XMLSchema">
 ...
</xsd:schema>
```

- Schema in a separate XML Document
- Vocabulary of Schema defined in special Namespace Alias „xsd“ is common
- There is a Schema for Schemas (don't worry!)
- „schema“ Element is always the Root

Prof. Dr. Thomas D. Schuster, Institute of Information Systems, ETH Zürich

ETH  
 Eidgenössische Technische Hochschule Zürich  
 Swiss Federal Institute of Technology Zürich

### Example Schema

```
<xsd:element name="book" type="BookType"/>
```

- „element“ in order to declare elements
- „name“ defines the name of the element.
- „type“ defines the type of the element
- Declarations under „schema“ are **global**
- Global element declarations are potential roots
- Example: „book“ is the only global element, root element must be a „book“.

Prof. Dr. Thomas D. Schuster, Institute of Information Systems, ETH Zürich

ETH  
 Eidgenössische Technische Hochschule Zürich  
 Swiss Federal Institute of Technology Zürich

### Example Schema

```
<xsd:complexType name="BookType">
 <xsd:sequence>
 ...
 </xsd:sequence>
</xsd:complexType>
```

- User-defined complex type
- Defines a sequence of sub-elements
- Attribute „name“ is name of Type
- This Typedefinition is **global**. Type can be used everywhere.

Prof. Dr. Thomas D. Schuster, Institute of Information Systems, ETH Zürich

ETH  
 Eidgenössische Technische Hochschule Zürich  
 Swiss Federal Institute of Technology Zürich

### Example Schema

```
<xsd:sequence>
 <xsd:element name="title" type="xsd:string"/>
</xsd:sequence>
```

- Local element declaration within a complex type
- („title“ cannot be root element of documents)
- „name“ and „type“ as before
- „string“ is built-in type of XML Schema

Prof. Dr. Thomas D. Schuster, Institute of Information Systems, ETH Zürich

ETH  
 Eidgenössische Technische Hochschule Zürich  
 Swiss Federal Institute of Technology Zürich

### Example Schema

```
<xsd:element name="author" type="PersonType"
 minOccurs="1" maxOccurs="unbounded"/>
```

- Local element declaration
- „PersonType“ is user-defined type
- „minOccurs“, „maxOccurs“ specify cardinality of „author“ Elements in „BookType“.
- Default: minOccurs=1, maxOccurs=1

Prof. Dr. Thomas D. Schuster, Institute of Information Systems, ETH Zürich

ETH  
 Eidgenössische Technische Hochschule Zürich  
 Swiss Federal Institute of Technology Zürich

### Example Schema

```
<xsd:element name="publisher" type="xsd:anyType"/>
```

- Local element declaration
- Every book has exactly one „publisher“ minOccurs, maxOccurs by default 1
- „anyType“ is built-in Type
- „anyType“ allows any content
- „anyType“ is default type. Equivalent definition: <xsd:element name="publisher" />

Prof. Dr. Thomas D. Schuster, Institute of Information Systems, ETH Zürich

**ETH**  
 Eidgenössische Technische Hochschule Zürich  
 Swiss Federal Institute of Technology Zürich

## Overview of Built-In Types

- **Numeric Values**  
 Integer, Short, Decimal, Float, Double, HexBinary, ...
- **Date, Timestamps, Durations**  
 Duration, DateTime, Time, Date, gMonth, ...
- **Text data**  
 String, NMTOKEN, NMTOKENS, NormalizedString
- **Other**  
 QName, AnyURI, ID, IDREFS, Language, Entity, ...
- **Overall 44 types**

http://www.w3.org/TR/xmlschema-1/

**ETH**  
 Eidgenössische Technische Hochschule Zürich  
 Swiss Federal Institute of Technology Zürich

## Definition of keys

- Part of element declaration
- Special sub element „key“
  - Describes context in which unique values are required (*selector*)
  - Describes the key (*field*)
  - Composite Keys by using multiple „field“
- Selector und fields: XPath (next section)
- Document validation with keys
  - Evaluate „selector“- result: set of nodes
  - Evaluate „fields“ on result aus: set of tuples
  - Check for duplicates in set of tuples

http://www.w3.org/TR/xmlschema-1/

**ETH**  
 Eidgenössische Technische Hochschule Zürich  
 Swiss Federal Institute of Technology Zürich

## Syntax of Key Definition

- Books are unique in „bib“ using „isbn“
 

```

 <element name = „bib“ > <complexType> <sequence>
 <element book maxOccurs = „unbounded“
 <complexType> <sequence> ... </sequence>
 <attribute name = „isbn“ type = „string“ />
 </complexType> </element> </sequence>
 <key name = „constraintX“ >
 <selector xpath = „book“ /> !! Get all books
 <field xpath = „@isbn“ /> !! Get isbn
 </key>
 </complexType> </element>

```

http://www.w3.org/TR/xmlschema-1/

**ETH**  
 Eidgenössische Technische Hochschule Zürich  
 Swiss Federal Institute of Technology Zürich

## References (Foreign Keys)

- Also part of element declaration
- Also „selector“ and „field(s)“
  - Selector describes which part should be checked for referential integrity
  - „field“ declarations compose „foreign key“
- Syntax (in our previous example):
 

```

 <keyref name = „constraintY“ refer = „constraintX“ >
 <selector xpath = „book/references“ />
 <field xpath = „@isbn“ />
 </keyref>

```

http://www.w3.org/TR/xmlschema-1/

**ETH**  
 Eidgenössische Technische Hochschule Zürich  
 Swiss Federal Institute of Technology Zürich

## Validate Document

```

 <?xml version=„1.0“>
 <book>
 <title>Die Wilde Wutz</title>
 <author><vorname>D.</vorname>
 <nachname>K.</nachname></author>
 <publisher> Addison Wesley,
 <state>CA</state>, USA
 </publisher>
 </book>

```

http://www.w3.org/TR/xmlschema-1/

**ETH**  
 Eidgenössische Technische Hochschule Zürich  
 Swiss Federal Institute of Technology Zürich

## Valid Document?

```

 <?xml version=„1.0“>
 <book>
 <title>Die Wilde Wutz</title>
 <author><vorname>D.</vorname>
 <nachname>K.</nachname></author>
 <publisher> Addison Wesley,
 <state>CA</state>, USA
 </publisher>
 </book>

```

http://www.w3.org/TR/xmlschema-1/

## Schema Validation

- Conformance Test
  - Result: „true“ or „false“
- Infoset Contribution (=> next chapter)
  - Annotate Types
  - Set Default Values
  - Result: new instance of the data model
- Tools: Xerces (Apache)
- Theory: Graph Simulation Algorithms
- Validation is a-posteri; explicit - not implicit!

## Additional aspects of schema

- Derivation/Description of types via „Facets“
  - Min/MaxValue
  - Patterns
  - List of possible Values
- Composition of complex types:
  - List
  - Union/Choice
- Derivation/Inheritance (complex, fine-grained)
- ...

## Document structure/type summary

- DTD
  - Relatively simple
  - Well understood
  - No information about types
- XML Schema
  - Very expressive
  - Complex
  - Need tool support
- A lot of XML document uses neither!