Einführung in XML

Markus Stocker

5. März 2018

Wer hat von XML schon gehört?

Warum das Symbol </>?

http://www.iconarchive.com/artist/hopstarter.html

Was ist XML?

- XML steht für e**X**tensible **M**arkup **L**anguage
- Auf Deutsch: Erweiterbare Auszeichnungssprache
- Es ist eine *markup language*, eine Auszeichnungssprache
- Eine Sprache zur Auszeichnung von Text in Dokumenten
- Ursprünglich aus typographie, z.B. Farbauszeichnung
- *Markup* ist syntaktisch unterscheidbar von Inhalt
- HTML ist wohl die bekannteste Auszeichnungssprache
- Markup Beispiele

```
XML: <planet>Earth</planet>
HTML: <h2>Earth</h2>
LaTeX: \textbf{Earth}
```

Was is XML?

- XML ist eine *meta* Auszeichnungssprache
  - Eine Sprache zur Erstellung anderer Auszeichnungssprachen
  - XML basierte Auszeichnungssprachen: SVG, XHTML, RSS, ...
- XML ist eine *beschreibende* Auszeichnungssprache
  - Reine Markierung von Inhalten
  - Keine Verarbeitungsanweisungen, z.B. Darstellung von Inhalten

Was is XML?

- Hierarchisch (semi-) strukturierte Daten
- Organisiert als XML Dokument
- Ein XML Dokument ist eine Zeichenfolge
- Strukturierte Daten als Textdatei
- XML ist Menschen und Maschinen lesbar
- Texteditor wie Notepad genügt um XML zu lesen und schreiben
- XML ist selbstbeschreibend
- XML is plattform- und implementationsunabhängig
- Standard für viele Office Produkte, z.B. LibreOffice

Beispiel: XML Dokument

```
<planets>
  <planet>Mercury</planet>
  <planet>Venus</planet>
  <planet>Earth</planet>
  <planet>Mars</planet>
  <planet>Jupiter</planet>
  <planet>Saturn</planet>
  <planet>Uranus</planet>
  <planet>Neptune</planet>
  <planet>Pluto</planet>
</planets>
```

Beispiel: SVG

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg version="1.1"
        id="Layer_1"
        xmlns="http://www.w3.org/2000/svg"
        xmlns:xlink="http://www.w3.org/1999/xlink"
        x="0px" y="0px" width="510px" height="510px"
        viewBox="0 0 510 510"
        enable-background="new 0 0 510 510"
        xml:space="preserve">

        <rect x="5" y="5" fill="none" stroke="#000000"
                stroke-width="10" stroke-miterlimit="10"
                width="500" height="500"/>
</svg>
```

Warum XML?

- Verbreitete Sprache
- Oft benutzt um Daten zwischen Anwendungen auszutauschen
- Insbesondere im Datenaustausch über das Internet (Web)
- XML Kenntnisse in ICT wichtig
- Heute keine Auszeichnung mehr, wird einfach angenommen

Lehrveranstaltung BIM-108-01

- Bachelor-Studiengang Informationsmanagement
- Pflichtmodul BIM-108—Datenstrukturierung (2. Semester)
- Lehrveranstaltung BIM-108-01—Grundlagen XML und RDF
- Keine Voraussetzungen
- Fragen: `markus.stocker@gmail.com`
- Ansprechperson HSH: Prof. Dr. Christian Wartena

`http://infom.wp.hs-hannover.de/wp-content/uploads/2017/08/StudienhandbuchBIMA5.pdf`

Veranstaltungsübersicht

	Tag	Thema
1	5. März	Einführung in XML
2	12. März	XML: Fortgeschrittene Themen
3	19. März	XPath
4	26. März	Schema: Document Type Definition (DTD)
5	9. April	Schema: XML Schema
6	16. April	Einführung in RDF
7	23. April	RDF Syntax: Eine breite Wahl
8	30. April	RDF: Fortgeschrittene Themen
9	7. Mai	SPARQL: Die RDF Abfragesprache
10	14. Mai	SPARQL: Fortgeschrittene Themen
11	28. Mai	Einführung in RDF Schema
12	4. Juni	Ontologien mit RDF Schema
13	11. Juni	Tools für RDF
14	18. Juni	XML und RDF: Rückschau und Ausblick
15	(?) 25. Juni	Klausur

Übungen

- Jede Vorlesung wird mit Übungen begleitet
- Übungen werden als Jupyter Notebooks durchgeführt
- Mehr dazu gleich in der ersten Übung

Prüfungsform

- Noch nicht festgelegt, mehr Information im April
- Voraussichtlich 2-stündige schriftliche Klausur
- Zusammengelegt mit BIM-108-02 (Inhaltserschliessung I - Methoden)
- Termin voraussichtlich am 25. Juni

Literatur

- Margit Becher. XML: DTD, XML-Schema, XPath, XQuery, XSLT, XSL-FO, SAX, DOM. 2009, Springer (Deutsch)
- Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph. Foundations of Semantic Web Technologies. 2010, CRC Press. (Englisch)
- Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph, York Sure. Semantic Web: Grundlagen. 2008, Springer. `https://doi.org/10.1007/978-3-540-33994-6` (Deutsch)
- `https://www.w3schools.com`
- Weitere online Ressourcen

Übersicht

- Geschichte
- Sprachkonstrukte
- Wohlgeformtheit

XML: Geschichte

- Entwicklung GML (Goldfarb, Mosher und Lorie, IBM) in 70er Jahren
- Sprache zur Auszeichnung von technischen Dokumenten
- Daraus wurde die Standard Generalised Markup Language (ISO, 1986)
- Spezifikation zur Definition von Auszeichnungssprachen
- HTML wurde die bekannte Anwendung von SGML
- Die Verbreitung von HTML führte zu Probleme
- Wie z.B. unterschiedliche Interpretation in Browsern
- Nicht geeignet für Speicherung und Austausch von Daten
- Entwicklung XML, zweite Hälfte 90er Jahren
- Seit 1998 eine W3C Recommendation
- Wie SGML eine Metasprache aber einfacher zu implementieren
- Anders als HTML, strenge Einhaltung der Sprachregeln

XML Sprachkonstrukte

- *Comment*
- *Tag*
- *Element*
- *Attribute*
- *Document*
- *Namespace*

Sprachkonstrukte: *Comment*

- Kommentare sind erlaubt
- Diese stehen zwischen den Zeichen `<!--` und `-->`
- Beispiel: `<!-- Das ist ein Kommentar -->`
- Kommentare dürften die Zeichen `--` nicht enthalten

Sprachkonstrukte: *Tag*

- Ein *tag* beginnt mit Zeichen < und endet mit Zeichen >
- Beispiel: `<planet>`
- Es gibt drei *tag* Arten
  - Das *opening tag*, z.B. `<planet>`
  - Das *closing tag*, z.B. `</planet>`
  - Das *empty-element tag*, z.B. `<planet/>`
- Beachte *tag* Gross- und Kleinschreibung
- Die *tags* `<planet>` und `<Planet>` sind nicht gleich

Sprachkonstrukte: *Element*

- Ein *element* (Element) ist ein Objekt welches
  - Mit einem *opening tag* beginnt
  - Mit einem **entsprechenden** *closing tag* endet
- Beispiel: `<planet>Earth</planet>`
- Bedenke Gross- und Kleinschreibung
- *Closing tag* `</Planet>` entspricht nicht dem *opening tag* `<planet>`
- Ein Element kann leer sein
- Beispiel: `<planet></planet>` oder Kurzform `<planet/>`

Sprachkonstrukte: *Element*

- Ein Element beinhaltet Text oder andere Elemente
- Beispiele

```
<!-- Element containing text -->
<planet>Earth</planet>

<!-- Element containing other elements -->
<planets>
  <planet>Earth</planet>
  <planet>Mars</planet>
</planets>
```

Sprachkonstrukte: *Element*

- Die Zeichen < und & sind in Text nicht erlaubt
- Diese müssen als `<` und `&` entsprechend kodiert werden
- Zudem sollten die Zeichen >, ’, und " ebenfalls kodiert werden
- Alternative: `<![CDATA[ ... ]]>`
- Beispiele

```
<planet>Mars radius is &lt; that of Earth</planet>

<planet>Earth is &quot;the only place where
  life is known to exist&quot;</planet>

<planet><![CDATA[
  Mars radius is < that of Earth
]]></planet>
```

Sprachkonstrukte: *Element*

- Elemente müssen korrekt verschachtelt werden
- Beispiel

```
<!-- Korrekte Verschachtelung -->
<planets><planet>Earth</planet></planets>

<!-- Inkorrekte Verschachtelung -->
<planets><planet>Earth</planets></planet>
```

Sprachkonstrukte: *Attribute*

- *Opening* und *empty-element tags* können Attribute enthalten
- Diese stehen innerhalb der < und > Klammern
- Als `name="value"` Paare
- Wobei `name` im *tag* eindeutig sein muss
- Attribute enthalten für das Element relevante Daten
- Beispiel

```
<planet name="Earth" radius="6371 km"/>
```

Sprachkonstrukte: *Element* oder *Attribute*

- Es gibt keine Regeln zur Verwendung von *element* oder *attribute*
- Das folgende Beispiel enthält die gleiche Information
- Wobei `name` und `radius` anders verwendet werden

```
<planet>
  <name>Earth</name>
  <radius>6371 km</radius>
</planet>

<planet name="Earth" radius="6371 km"/>
```

Sprachkonstrukte: *Element* oder *Attribute*

- Ein Element kann mehrere Werte enthalten und ist erweiterbar

```
<planet>
  <radius>6371 km</radius>
</planet>

<planet>
  <radius>
    <length>6371</length>
    <unit>km</unit>
  </radius>
</planet>
```

Sprachkonstrukte: *Document*

- Ein XML *document* (Dokument) ist ein Textdokument
- Beginnt mit einer (optionalen) *declaration*
- Keine Leerzeichen davor
- Versionsnummer und (optional) die Kodierung (meist `utf-8`)
- Ein XML Dokument muss genau ein (äusserstes) Element enthalten
- Dieses wird *root element* (Wurzelelement) genannt
- Beispiel

```
<?xml version="1.0" encoding="utf-8"?>
<planets/>
```

Sprachkonstrukte: *Namespace*

- Vermeidung von Namenskonflikte
- Beispiel: `<planet>` der Himmelskörper oder die Software?
- Diese haben meist unterschiedliche Attribute
- Verwendung von Namespräfix
- Beispiel: `<astro:planet>` und `<soft:planet>`
- Bedarf der Definition von *namespaces* (Namensräume)
- Namensraum Definition als Attribut in (Wurzel-) Element
- Beispiel

```
<planets>
  <astro:planet xmlns:astro="http://astronomy.org">
    <astro:name>Earth</astro:name>
  </astro:planet>
</planets>
```

Wohlgeformtheit

- Ist ein XML Dokument syntaktisch korrekt ist es wohlgeformt
- Wohlgeformtheit (*well-formed*) erwartet, unter anderem
  - Genau ein Wurzelelement
  - Elemente müssen korrekt strukturiert sein
  - Korrekte Verschachtelung der Elemente

Zusammenfassung

- XML ist eine erweiterbare Auszeichnungssprache
- Metasprache zur Spezifikation von Auszeichnungssprachen
- Wichtige Sprachkonstrukte: *Tag*, *Element*, *Attribute*, *Document*
- Wichtiges Konzept: Wohlgeformtheit
- Weit verbreitet, insb. im Datenaustausch (Web)
- XML Kenntnisse in ICT generell vorausgesetzt