XML: Fortgeschrittene Themen

Markus Stocker

12. März 2018

Rekapitulation: Das ist ein ...

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Yet another example -->
<a:planets xmlns:a="http://astronomy.org">
<a:planet radius="6371 km">Earth</a:planet>
<a:planet><![CDATA[Mars]]></a:planet>
</a:planets>
```

Rekapitulation: Das ist eine ...

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Yet another example -->
<a:planets xmlns:a="http://astronomy.org">
<a:planet radius="6371 km">Earth</a:planet>
<a:planet><![CDATA[Mars]]></a:planet>
</a:planets>
```

Rekapitulation: Das ist ein ...

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Yet another example -->
<a:planets xmlns:a="http://astronomy.org">
<a:planet radius="6371 km">Earth</a:planet>
<a:planet><![CDATA[Mars]]></a:planet>
</a:planets>
```

Rekapitulation: Das ist ein ...

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Yet another example -->
<a:planets xmlns:a="http://astronomy.org">
<a:planet radius="6371 km">Earth</a:planet>
<a:planet><![CDATA[Mars]]></a:planet>
</a:planets>
```

Rekapitulation: Das ist ein ...

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Yet another example -->
<a:planets xmlns:a="http://astronomy.org">
<a:planet radius="6371 km">Earth</a:planet>
<a:planet><![CDATA[Mars]]></a:planet>
</a:planets>
```

Rekapitulation: Das ist ein ...

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Yet another example -->
<a:planets xmlns:a="http://astronomy.org">
<a:planet radius="6371 km">Earth</a:planet>
<a:planet><![CDATA[Mars]]></a:planet>
</a:planets>
```

Rekapitulation: Das ist ein ...

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Yet another example -->
<a:planets xmlns:a="http://astronomy.org">
<a:planet radius="6371 km">Earth</a:planet>
<a:planet><![CDATA[Mars]]></a:planet>
</a:planets>
```

Rekapitulation: Das ist ein ...

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Yet another example -->
<a:planets xmlns:a="http://astronomy.org">
<a:planet radius="6371 km">Earth</a:planet>
<a:planet><![CDATA[Mars]]></a:planet>
</a:planets>
```

Rekapitulation: Das ist ...

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Yet another example -->
<a:planets xmlns:a="http://astronomy.org">
<a:planet radius="6371 km">Earth</a:planet>
<a:planet><![CDATA[Mars]]></a:planet>
</a:planets>
```

Dieses XML Dokument ist nicht wohlgeformt weil ...

```
<a:planets xmlns:a="http://astronomy.org">
<a:planet radius="6371 km">Earth < Mars</a:Planet>
<a:planet>![CDATA[Mars is ]]> next frontier]]></a:planets>
</a:planet> <!-- Yet another -- example -->
```

Übersicht

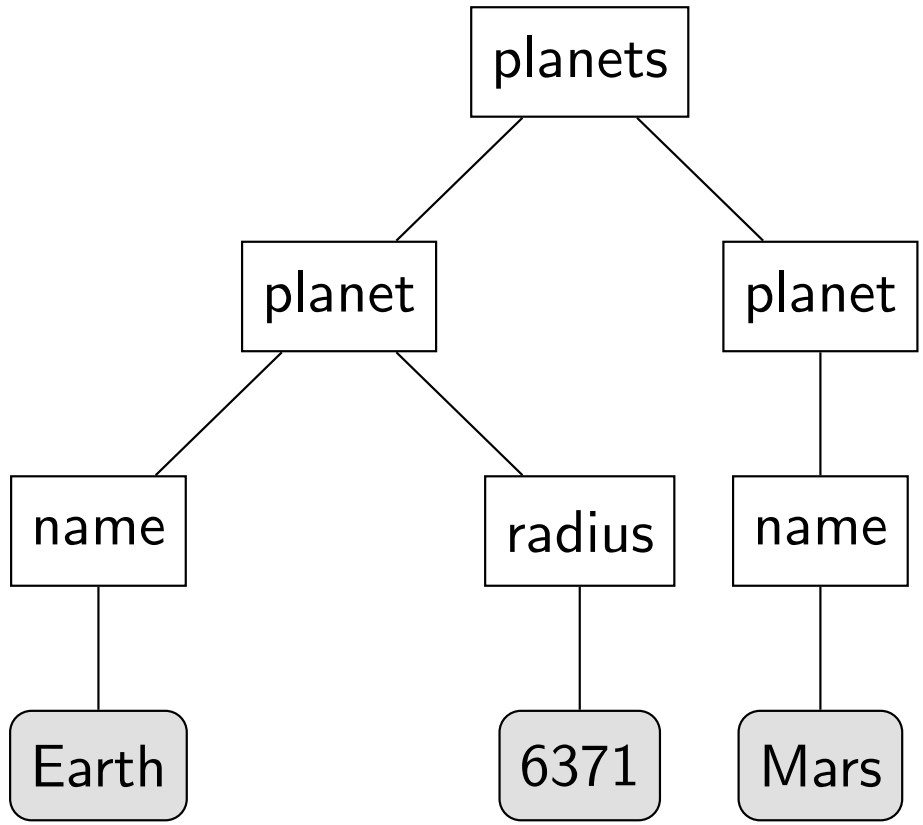
- XML Baumstruktur
- Sich auf einen Standard festlegen: Vokabularien in XML
- XML Vokabularien und Bedeutung
- XML in der Programmierung: Lesen, Schreiben, Verarbeiten
- XML und Datenbanken
- XML und Web Services
- Einige Nachteile von XML

Baumstruktur

- Die Struktur der Elemente eines XML Dokumentes ergibt ein Baum
- Dokument enthält ein einziges *root element*
- Das Element an der Spitze des Baumes
- Element kann ein oder mehrere “Kindelemente” (*child*) enthalten
- Diese sind dem “Elternelement” (*parent*) untergeordnet

Baumstruktur

```
<planets>
  <planet>
    <name>Earth</name>
    <radius>6371</radius>
  </planet>
  <planet>
    <name>Mars</name>
  </planet>
</planets>
```



Vokabularien in XML

- Dass Daten in XML vorliegen bedeutet nicht, dass diese nützlich sind
- Programme A und B können Zeichnungen als XML speichern
- Bedeutet nicht, dass die Zeichnungen austauschbar sind
- Austauschbarkeit bedarf gemeinsame Festlegung auf ein Vokabular
- Auch “Standard” genannt, z.B. Scalable Vector Graphics (SVG)

Vokabularien in XML

- Sich auf einen Vokabular festlegen bedeutet
- Entweder ein existierendes aufnehmen (z.B. einen Standard)
- Oder zumindest darauf aufbauen
- Oder aber mit der community ein Vokabular erstellen

Vokabularien in XML

- Wie man Daten in XML strukturiert ist entscheidend
- Eine Anwendung erwartet eine gewisse Struktur
- Anwendungen werden anhand einer Struktur entwickelt

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

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

Vokabularien in XML

- *Tags* haben für Software keinerlei Bedeutung
- Ob `<planet>Earth</planet>` oder `<x>Earth</x>`
- Für Software macht dies kein (grosser) Unterscheid
- Nur die Baumstruktur ist für die Software von Bedeutung
- Selbstbeschreibend ist XML also hauptsächlich für Menschen
- Jedoch ist XML selbst für Menschen oft nicht einfach zu verstehen
- Weil die Bedeutung der *tags* oft nicht klar ist

XML in der Programmierung: Lesen und Schreiben

- XML ist in gängigen Programmiersprachen les- und schreibar
- Und zwar aus den verschiedesten Sourcen
- Lesen aus `string` haben wir bereits gesehen
- Möglich ist auch das Lesen und Schreiben aus/zu
  - Dateien
  - Datenbanken
  - Internet

XML in der Programmierung: Programmatisch Schreiben

```
from lxml import etree as et

planets = et.Element('planets')
planet_earth = et.SubElement(planets, 'planet')
planet_mars = et.SubElement(planets, 'planet')
earth_name = et.SubElement(planet_earth, 'name')
earth_radius = et.SubElement(planet_earth, 'radius')
mars_name = et.SubElement(planet_mars, 'name')
earth_name.text = 'Earth'
earth_radius.text = '6371'
mars_name.text = 'Mars'

print(et.tostring(planets, pretty_print=True).decode('utf-8'))
```

```
<planets>
  <planet>
    <name>Earth</name>
    <radius>6371</radius>
  </planet>
  <planet>
    <name>Mars</name>
  </planet>
</planets>
```

XML in der Programmierung: Verarbeiten

- Einmal gelesen, kann XML programmatisch verarbeitet werden
- Programmiersprachen stellen dafür Funktionalität zur Verfügung
- Diese erlaubt das Durchlaufen des Baumes
- Wie auch der geziehlte Zugriff auf Elemente, Attribute, etc.

XML in der Programmierung: Verarbeiten

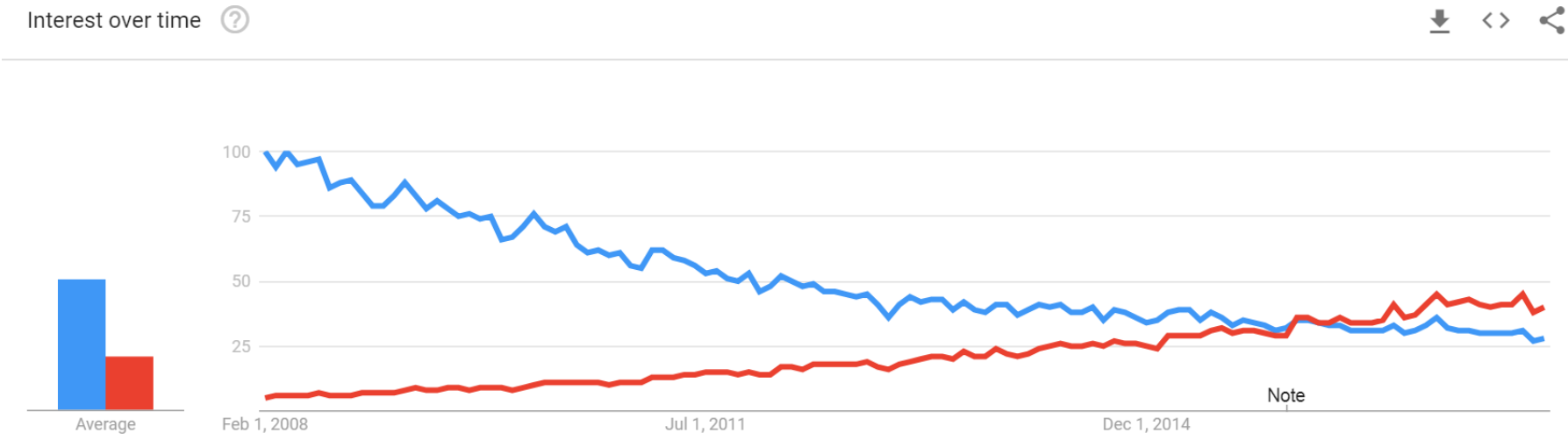
```
from lxml import etree as et

planets = et.fromstring("""
  <planets>
    <planet>
      <name>Earth</name>
      <radius>6371</radius>
    </planet>
    <planet>
      <name>Mars</name>
    </planet>
  </planets>
""")

for planet in planets:
  for quality in planet:
    print('{}: {}'.format(quality.tag, quality.text))

name: Earth
radius: 6371
name: Mars
```

XML und Datenbanken

- Datenbank die XML Dokumente speichern und durchsuchen zu kann
- Dokumentenorientierten Datenbank (NoSQL)
- Underscheidet sich von relationalen Datenbankmodellen
- XPath, XQuery, XSL zur Abfrage und Manipulation verwendet
- Native XML-Datenbanksysteme: BaseX, Berkeley DB XML, etc.
- XML-enabled Datenbanken: IBM, Oracle, Microsoft

XML und Web Services

- XML besonders im Internet Datenaustausch verbreitet
- Viele Web Services and Web API liefern Daten in XML
- Allerdings nimmt die Bedeutung anderer Formate zu (z.B. JSON)



Google Trends: XML (blau) und JSON (rot), seit 2008

XML und Web Services

```
import requests

url = '{}?verb={}&metadataPrefix={}&identifier={}'.format(
        'http://ws.pangaea.de/oai/provider',
        'GetRecord',
        'datacite3',
        'oai:pangaea.de:doi:10.1594/PANGAEA.858171'
)

r = requests.get(url)

x = et.XML(bytes(bytearray(r.text, 'utf-8')))

print(x.find(
  './/{http://datacite.org/schema/kernel-3}identifier[@identifierType="DOI"]'
  ).text)

10.1594/PANGAEA.858171
```

Einige Nachteile von XML

- Das `tagging` ist “Ballast”
- Speziell bei stark regulären Daten (z.B. Tabellen)
- Grosser Speicherbedarf und Bandbreite
- Problematisch wenn diese fehlen, z.B. Sensordaten

Zusammenfassung