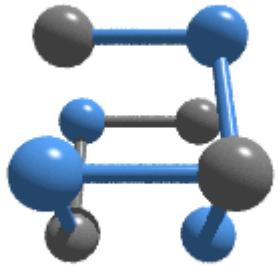


Präsenzveranstaltung zur E-Learning-Veranstaltung

Einführung in XML

Sommersemester 2008

Prof. Dr. Klaus-Peter Fähnrich
Dr. Maik Thränert

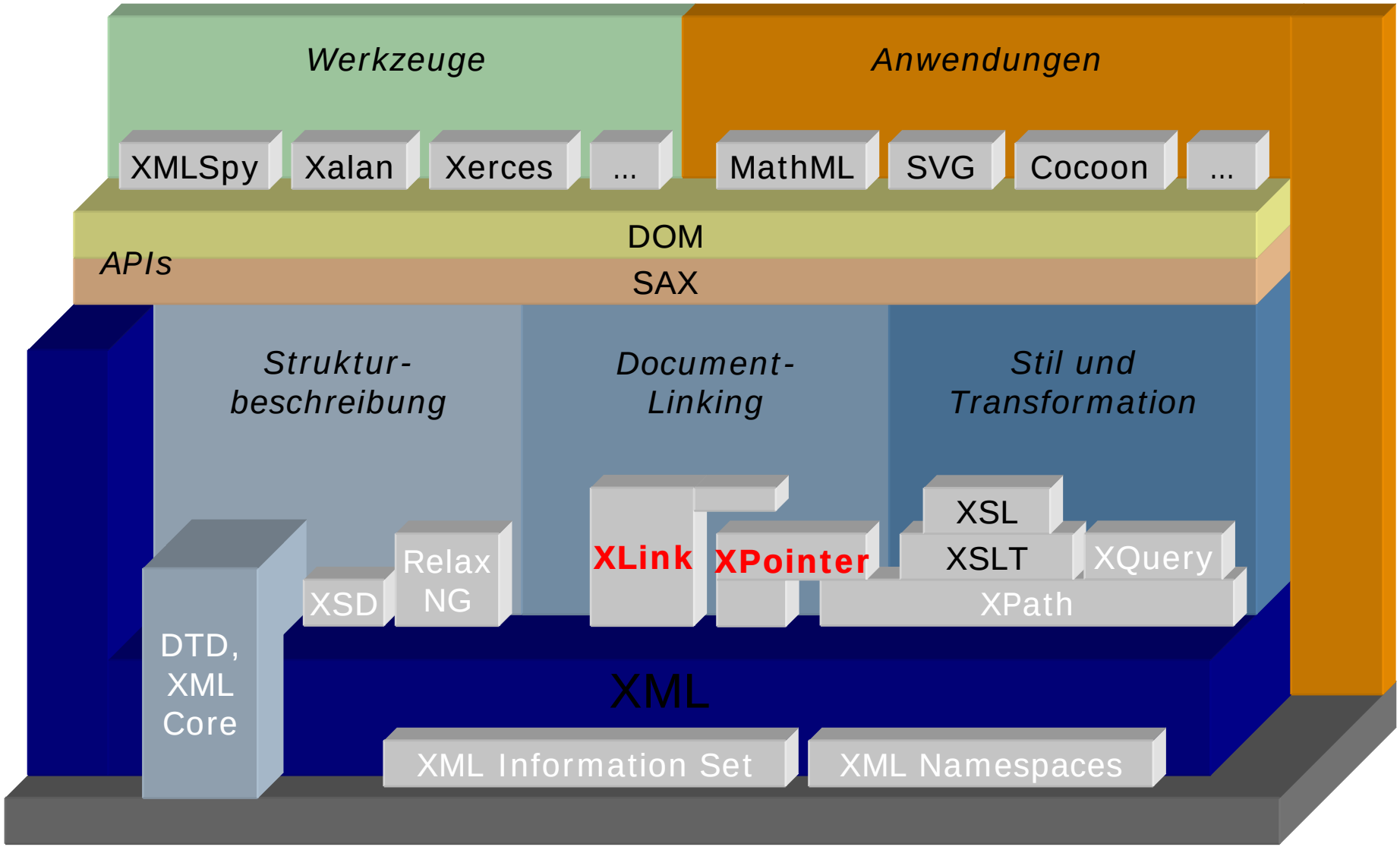
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Agenda

- **XPointer**
- XLink
- XSLT
- XSL-FO
- Fragen?
- Weiterer Ablauf

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Gliederung der Vorlesung



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Lernziele

- Aufgaben und Aufbau von XPointer kennen
  - Datentypen
  - Funktionen
- Aufgaben und Typen von XLinks kennen
  - einfacher Link
  - erweiterter Link
  - resource, locator, arc etc.
- Link-Implementierungen von HTML und XLink vergleichen können

Gliederung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

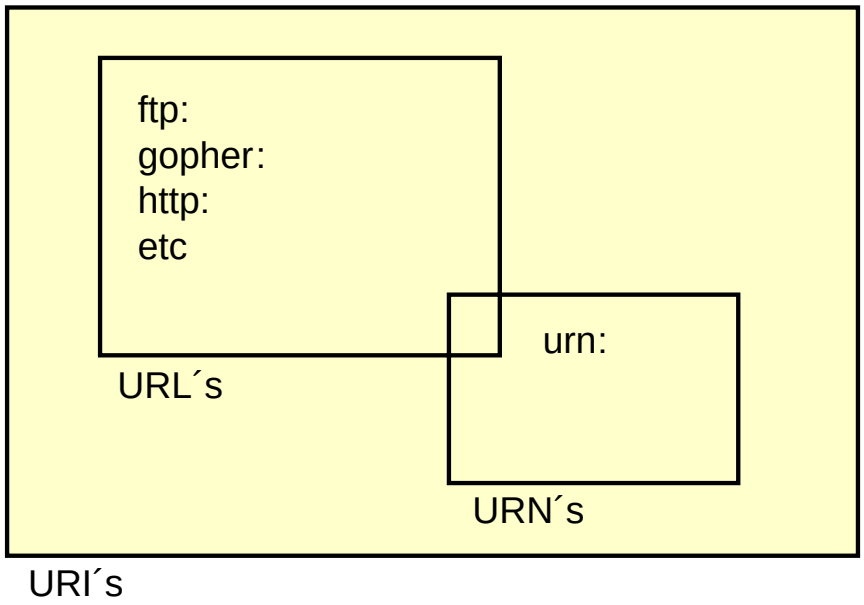
1. **URI/URN/URL**
   - Überblick,
   - URI
2. Das XPointer-Framework
   - Spezifikationen,
   - xpointer(),
   - element()
3. XLink
   - Motivation,
   - Anatomie eines XLinks,
   - Arten von Links,
   - Elemente und Attribute für XLink

1. URI – Überblick

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

rfc2396.txt

UR*



Aus RFC 2396

Gliederung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1. URI/URN/URL
   - Überblick,
   - URI
2. **Das XPointer-Framework**
   - Spezifikationen,
   - xpointer(),
   - element()
3. XLink
   - Motivation,
   - Anatomie eines XLinks,
   - Arten von Links,
   - Elemente und Attribute für XLink

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. XPointer

- Die XPointer Spezifikation besteht aus 4 verschiedenen Dokumenten:
- XML Pointer Language W3C (Working Draft 16 August 2002):

1. XPointer Framework
   - W3C Recommendation 25 March 2003
2. XPointer element() Scheme
   - W3C Recommendation 25 March 2003
3. XPointer xmlns() Scheme
   - W3C Recommendation 25 March 2003
4. XPointer xpointer() Scheme
   - W3C **Working Draft** 19 December 2002

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. XPointer Framework

- XPointer Framework Syntax:

```
Pointer ::= Shorthand | SchemeBased
Shorthand ::= NCName
SchemeBased ::= PointerPart (S? PointerPart)*
PointerPart ::= SchemeName '(' SchemeData ')'
SchemeName ::= QName
SchemeData ::= EscapedData*
EscapedData ::= NormalChar | '^(' | '^)' | '^^' | '('
   SchemeData ')'
NormalChar ::= UnicodeChar - [()^]
UnicodeChar ::= [#x0-#x10FFFF]
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. XPointer element() Scheme

- Diese Spezifikation erlaubt die Benutzung grundlegender XPointer-Ausdrücke, um Elemente eines XML-Dokumentes zu adressieren,
- Mittels `element()` können einzelne Elemente eines XML-Dokumentes adressiert werden

- Beispiele:

`element(bla)`

- Es wird das Element mit der ID "bla" ausgewählt,

`element(/x/y/z)`

- Es wird das x-te Toplevelelement ausgewählt, davon dann das y-te Kindelement, von welchem wiederum das z-te Kind selektiert wird

`element(bla/2)`

- Es wird das zweite Kindelement des Elementes mit der ID "bla" ausgewählt,

Quelle: http://www.w3.org/TR/xptr-element/

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. XPointer xpointer() Scheme

- XPath ermöglicht den Zugriff auf Knoten bzw. Knotenmengen in einem XML-Dokument,
- Oftmals sind jedoch nur Teile eines Knotens von Bedeutung,
  - Beispielsweise möchte man auf einen Teil eines Textknoten zugreifen,
  - Oder auf einen Teil in einem Kommentar
  - Darauf mit XPath zuzugreifen ist nicht möglich,
- Um diese Zugriffe zu realisieren, wurde das xpointer() Schema entwickelt
  - basiert auf den von XPath bekannten Ausdrücken (`Expr`) und erweitert die dort vorgestellten Konzepte

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. XPointer xpointer() Scheme

- Das xpointer() Schema unterscheidet zwischen Location Types und Funktionen um auf diese Locations zugreifen zu können,

- **Location Types:**
  - Point-Location
  - Range-Location
  - Covering Ranges for All Location Types

- Funktionen:
  - range-to(),
  - string-range(),
  - covering-range(),
  - range-inside(),
  - start-point(),
  - end-point()

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. XPointer xpointer() Scheme

- Das xpointer() Schema unterscheidet zwischen Location Types und Funktionen um auf diese Locations zugreifen zu können,

- Location Types:
  - Point-Location
  - Range-Location
  - Covering Ranges for All Location Types

- **Funktionen:**
  - range-to(),
  - string-range(),
  - covering-range(),
  - range-inside(),
  - start-point(),
  - end-point()

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. Beispiel

adressdb.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<AdressDB  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="Beispiel1.xsd">
   <Adresse>
        <Person Anrede="Frau">
                <Vorname>Eva</Vorname>
                <Name>Mustermann</Name>
        </Person>
        <Strasse Nummer="4">Beispielstrasse</Strasse>
        <Ort PLZ="54783">Musterstadt</Ort>
        <Telefon>+49-(0)999-1234567</Telefon>
        <eMail>eva.mustermann@muster.mail</eMail>
   </Adresse>
   <Adresse>
        <Person Anrede="Herr">
                <Vorname>Hans</Vorname>
                <Name>Meier</Name>
        </Person>
        <Postfach>123456</Postfach>
        <Ort PLZ="80939"/>
        <Telefon>(089)9874563</Telefon>
        <eMail>hans@hans.meier.de</eMail>
   </Adresse>
</AdressDB>
```

Bereich

```
xpointer(start-point(//Person))
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. Beispiel

adressdb.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<AdressDB  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="Beispiel1.xsd">
   <Adresse>
        <Person Anrede="Frau">
                <Vorname>Eva</Vorname>
                <Name>Mustermann</Name>
        </Person>
        <Strasse Nummer="4">Beispielstrasse</Strasse>
        <Ort PLZ="54783">Musterstadt</Ort>
        <Telefon>+49-(0)999-1234567</Telefon>
        <eMail>eva.mustermann@muster.mail</eMail>
   </Adresse>
   <Adresse>
        <Person Anrede="Herr">
                <Vorname>Hans</Vorname>
                <Name>Meier</Name>
        </Person>
        <Postfach>123456</Postfach>
        <Ort PLZ="80939"/>
        <Telefon>(089)9874563</Telefon>
        <eMail>hans@hans.meier.de</eMail>
   </Adresse>
</AdressDB>
```

Bereich

```
xpointer(end-point(//Person))
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. Beispiel

adressdb.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<AdressDB  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="Beispiel1.xsd">
    <Adresse>
        <Person Anrede="Frau">
            <Vorname>Eva</Vorname>
            <Name>Mustermann</Name>
        </Person>
        <Strasse Nummer="4">Beispielstrasse</Strasse>
        <Ort PLZ="54783">Musterstadt</Ort>
        <Telefon>+49-(0)999-1234567</Telefon>
        <eMail>eva.mustermann@muster.mail</eMail>
    </Adresse>
    <Adresse>
        <Person Anrede="Herr">
            <Vorname>Hans</Vorname>
            <Name>Meier</Name>
        </Person>
        <Postfach>123456</Postfach>
        <Ort PLZ="80939"/>
        <Telefon>(089)9874563</Telefon>
        <eMail>hans@hans.meier.de</eMail>
    </Adresse>
</AdressDB>
```

1.

2.

Container-Node

Bereich

```
xpointer(range(//Adresse/Person))
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. Beispiel

adressdb.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<AdressDB  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="Beispiel1.xsd">
   <Adresse>
      <Person Anrede="Frau">
         <Vorname>Eva</Vorname>
         <Name>Mustermann</Name>
      </Person>
      <Strasse Nummer="4">Beispielstrasse</Strasse>
      <Ort PLZ="54783">Musterstadt</Ort>
      <Telefon>+49-(0)999-1234567</Telefon>
      <eMail>eva.mustermann@muster.mail</eMail>
   </Adresse>
   <Adresse>
      <Person Anrede="Herr">
         <Vorname>Hans</Vorname>
         <Name>Meier</Name>
      </Person>
      <Postfach>123456</Postfach>
      <Ort PLZ="80939"/>
      <Telefon>(089)9874563</Telefon>
      <eMail>hans@hans.meier.de</eMail>
   </Adresse>
</AdressDB>
```

1. 2.

Container-Node

Bereich

```
xpointer(range-inside(//Adresse/Person))
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. Beispiel

adressdb.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<AdressDB  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="Beispiel1.xsd">
   <Adresse>
        <Person Anrede="Frau">
                <Vorname>Eva</Vorname>
                <Name>Mustermann</Name>
        </Person>
        <Strasse Nummer="4">Beispielstrasse</Strasse>
        <Ort PLZ="54783">Musterstadt</Ort>
        <Telefon>+49-(0)999-1234567</Telefon>
        <eMail>eva.mustermann@muster.mail</eMail>
   </Adresse>
   <Adresse>
        <Person Anrede="Herr">
                <Vorname>Hans</Vorname>
                <Name>Meier</Name>
        </Person>
        <Postfach>123456</Postfach>
        <Ort PLZ="80939"/>
        <Telefon>(089)9874563</Telefon>
        <eMail>hans@hans.meier.de</eMail>
   </Adresse>
</AdressDB>
```

Bereich

```
xpointer(/AdressDB/Adresse/Person[position()=1]
/range-to(/AdressDB/Adresse/Postfach[position()=last()]))
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. Beispiel

```
<?xml version="1.0" encoding="UTF-8"?>
<AdressDB  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="Beispiel1.xsd">
   <Adresse>
        <Person Anrede="Frau">
                <Vorname>Eva</Vorname>
                <Name>Mustermann</Name>
        </Person>
        <Strasse Nummer="4">Beispielstrasse</Strasse>
        <Ort PLZ="54783">Musterstadt</Ort>
        <Telefon>+49-(0)999-1234567</Telefon>
        <eMail>eva.mustermann@muster.mail</eMail>
   </Adresse>
   <Adresse>
        <Person Anrede="Herr">
                <Vorname>Hans</Vorname>
                <Name>Meier</Name>
        </Person>
        <Postfach>123456</Postfach>
        <Ort PLZ="80939"/>
        <Telefon>(089)9874563</Telefon>
        <eMail>hans@hans.meier.de</eMail>
   </Adresse>
</AdressDB>
```

Bereich

```
xpointer(/AdressDB/Adresse/Person
/range-to(/AdressDB/Adresse/Person[position()=last()]))
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. Beispiel

adressdb.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<AdressDB  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="Beispiel1.xsd">
   <Adresse>
        <Person Anrede="Frau">
                <Vorname>Eva</Vorname>
                <Name>Mustermann</Name>
        </Person>
        <Strasse Nummer="4">Beispielstrasse</Strasse>
        <Ort PLZ="54783">Musterstadt</Ort>
        <Telefon>+49-(0)999-1234567</Telefon>
        <eMail>eva.mustermann@muster.mail</eMail>
   </Adresse>
   <Adresse>
        <Person Anrede="Herr">
                <Vorname>Hans</Vorname>
                <Name>Meier</Name>
        </Person>
        <Postfach>123456</Postfach>
        <Ort PLZ="80939"/>
        <Telefon>(089)9874563</Telefon>
        <eMail>hans@hans.meier.de</eMail>
   </Adresse>
</AdressDB>
```

Bereich

```
xpointer(string-range(//Ort, "stadt"))
xpointer(string-range(//*, "muster", 3, 4))
```

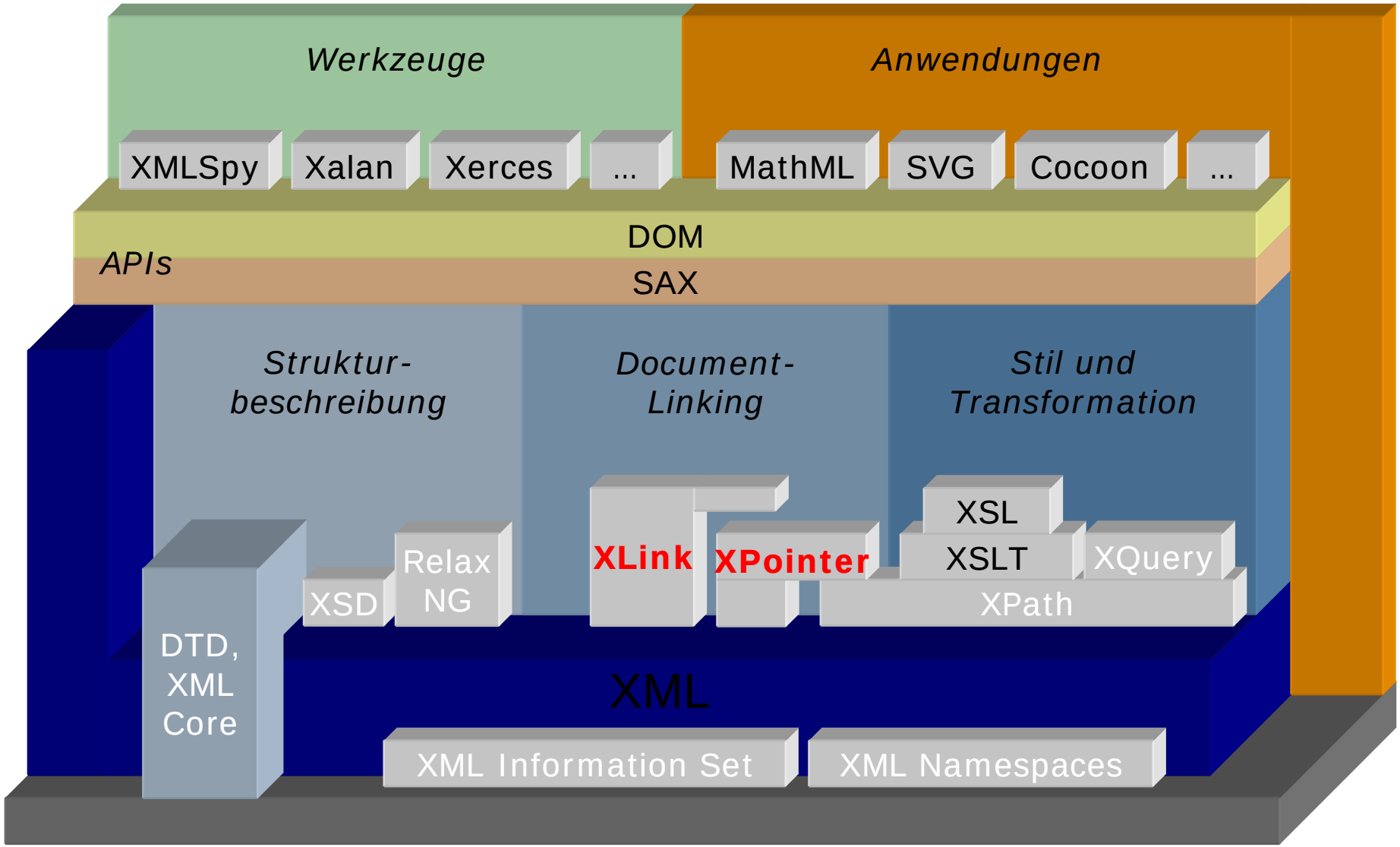
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Agenda

- XPointer
- **XLink**
- XSLT
- XSL-FO
- Fragen?
- Weiterer Ablauf

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Gliederung der Vorlesung



Gliederung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1. URI/URN/URL
   - Überblick,
   - URI
2. Das XPointer-Framework
   - Spezifikationen,
   - xpointer(),
   - element()
3. **XLink**
   - **Motivation**,
   - Anatomie eines XLinks,
   - Arten von Links,
   - Elemente und Attribute für XLink

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Motivation – Requirements by W3C

1. XLink must be straightforwardly usable over the Internet.
   - Es muss ein "offenes" System von Links unterstützt werden,
   - Es müssen Links in verschiedene Richtungen verarbeitet werden,
     - Unidirektionale Links,
     - Multidirektionale Links
   - Unterstützung von Internationalisierung ist wichtig,
   - Zusammenarbeit zwischen den einzelnen Applikationen muss gewährleistet sein

2. XLink must be usable by a wide variety of link usage domains and classes of linking application software.
   - XLink sollte für viele Bereiche einsetzbar sein und nicht nur einen bestimmten bevorzugen,
   - Es sollten auch nicht Browser vor anderen Applikationen bevorzugt werden

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Motivation – Requirements by W3C

3. XLink must support HTML 4.0 linking constructs.

4. The XLink expression language must be XML.
   - XLink-Strukturen müssen in XML ausgedrückt werden können

5. The XLink design must be prepared quickly.
   - XLink sollte so schnell wie möglich entwickelt werden

6. The XLink design must be formal, concise, and illustrative.
   - Das Design von XLink soll nicht die Linkstruktur mit der XML-Syntax, die den Link ausdrückt, vermischen.

7. XLinks must be human-readable and human-writable.
   - XLinks sollen von Menschen gelesen und geschrieben werden können

3. XLink – Motivation – Requirements by W3C

8. XLinks may reside within or outside the documents in which the participating resources reside.
   - Da XLinks die Probleme von HTML-Links lösen sollen, ist ein zufriedenstellender Mechanismus zur Behandlung von ausgehenden Links notwendig.
   - Ebenso sollen Links unterstützt werden, die außerhalb der zu verlinkenden Dokumente stehen.

9. XLink must represent the abstract structure and significance of links.

10. XLink must be feasible to implement.

11. XLink must be informed by knowledge of established hypermedia systems and standards.

Gliederung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1. URI/URN/URL
   - Überblick,
   - URI
2. Das XPointer-Framework
   - Spezifikationen,
   - xpointer(),
   - element()
3. **XLink**
   - Motivation,
   - **Anatomie eines XLinks,**
   - Arten von Links,
   - Elemente und Attribute für XLink

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Anatomie eines XLinks

- Bei HTML bestehen Links immer aus einem Ziel und sind somit immer ausgehende Links,

```
<a href="http://link.de/">link</a>
<img src="bild.gif"/>
```

- Eigenschaften von HTML-Links:
  - Nutzen URIs um ein Ziel anzugeben,
  - Link steht nur an einem der beiden Enden der Kante und zielt auf das andere Ende,
  - Der Nutzer kann den Link nur in einer Richtung verfolgen,
  - Wie der Link dargestellt wird hängt nur vom Browser ab, nicht vom Link selbst

Quelle: XLink-Spezifikation

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Anatomie

- Startpunkt:
  - Von hier startet der Link,
- Endpunkt:
  - An diese Stelle zeigt ein Link,
- Kante:
  - Die Verbindung zwischen Start- und Endpunkt,
  - Hat keine Richtung,
- Richtung:
  - Gibt die Richtung an, in der eine Kante verfolgt wird,
- Verhalten (actuate):
  - Gibt an, wie der Link beim Einlesen behandelt werden soll,
- Darstellung (show):
  - Falls das XML-Dokument angezeigt wird, gibt dieses Attribut an, wie das Ziel des XLink dargestellt wird

Gliederung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

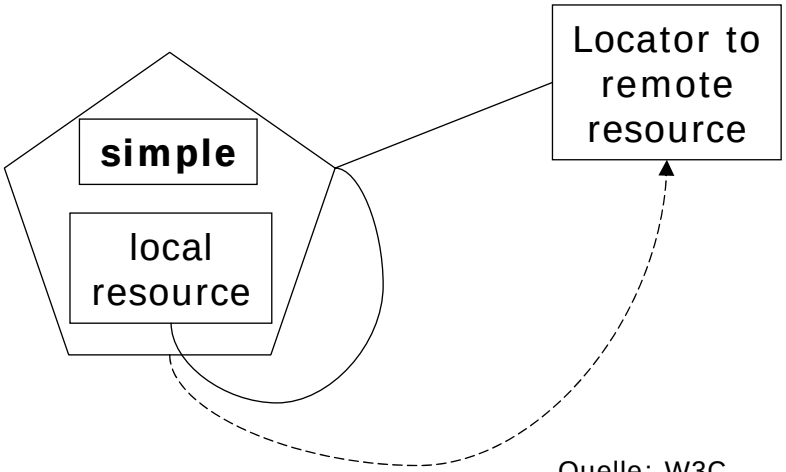
1. URI/URN/URL
   - Überblick,
   - URI
2. Das XPointer-Framework
   - Spezifikationen,
   - xpointer(),
   - element()
3. **XLink**
   - Motivation,
   - Anatomie eines XLinks,
   - **Arten von Links,**
   - Elemente und Attribute für XLink

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

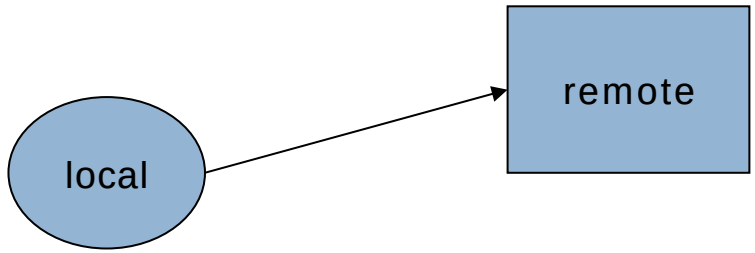
3. XLink – Einfache Links

- Definition: Ein einfacher Link ist ein Link, der genau zwei Ressourcen zueinander in Beziehung setzt, eine lokale und eine entfernte, mit einer Kante von der ersteren zur letzteren Ressource. Ein einfacher Link ist also immer ein ausgehender Link.

- folgendes Bild zeigt einen einfachen Link
  - assoziiert eine lokale und eine entfernte Ressource
  - bietet implizit eine einzige Traversierungskante von der lokalen Ressource zur entfernten



Quelle: W3C



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Einfache Links

- Einfache Verweise entsprechen den aus HTML bekannten Links

```
<a href="http://www.uni-leipzig.de">Uni Leipzig</a>
```

ist die aus (X)HTML bekannte Art von Links,

```
<ein-link xlink:type="simple"
   xlink:href="http://www.uni-leipzig.de">
   Uni-Leipzig</ein-link>
```

ist derselbe Link in XML als XLink

- Die Unterschiede zwischen HTML-Links und XLinks bestehen in:
  - dem Attribut "`xlink:type`" mit dem Wert "`simple`" und
  - im Namensraumpräfix "`xlink`",
  - jedes Element kann einen XLink beinhalten

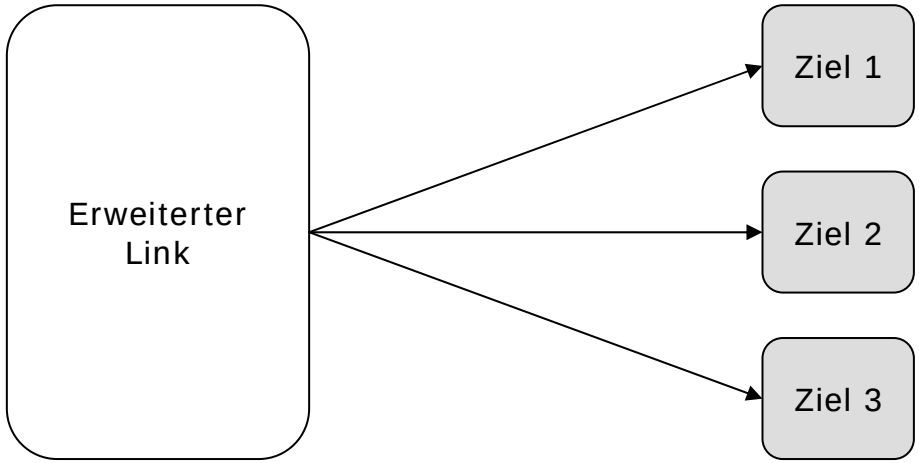
adressdb.xml

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Erweiterte Links

Erweiterte Links mit XLink:

- Im Gegensatz zu einfachen Links können erweiterte Links mehr als nur eine Ressource referenzieren.

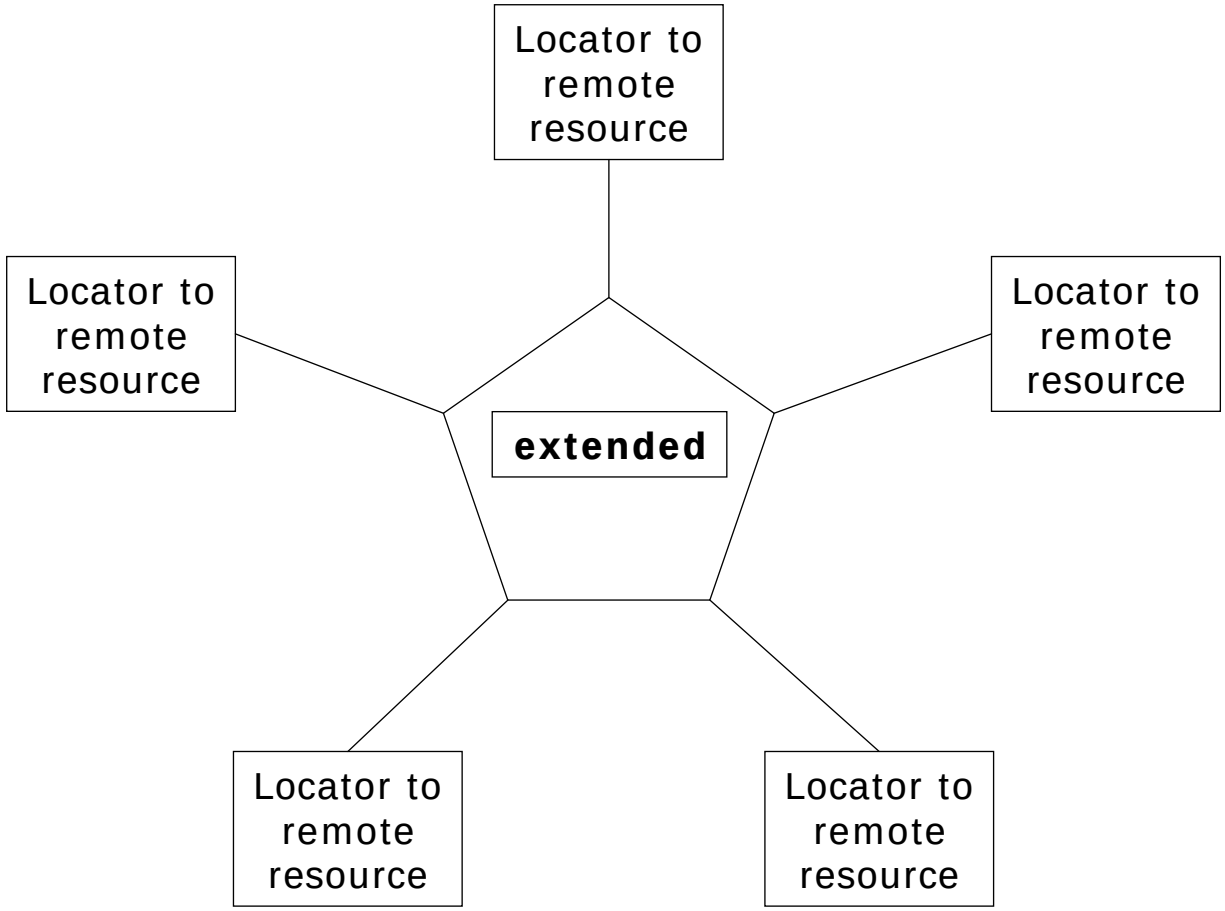


- Falls alle referenzierten Ressourcen entfernt sind, handelt es sich um einen "Out-of-line-Link",
- falls jedoch mindestens eine Ressource lokal ist, ist es ein "Inline-Link".

Quelle: Behme/Mintert

3. XLink – Erweiterte Links

Out-of-line-Link:



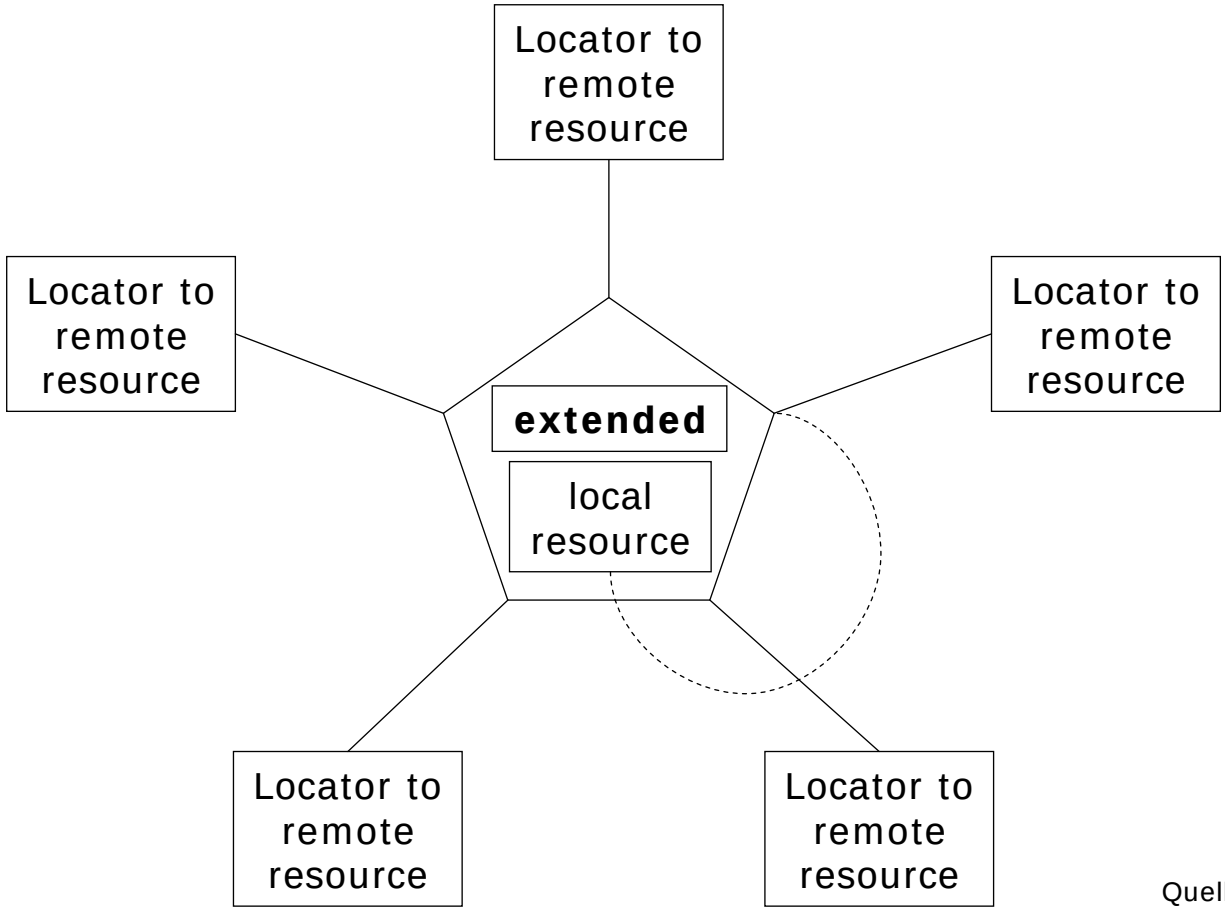
Quelle: W3C

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Erweiterte Links

Das folgende Diagramm zeigt einen erweiterten Link, der fünf entfernte und eine lokale Ressource assoziiert:

Inline-Link



Quelle: W3C

Gliederung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1. URI/URN/URL
   - Überblick,
   - URI
2. Das XPointer-Framework
   - Spezifikationen,
   - xpointer(),
   - element()
3. **XLink**
   - Motivation,
   - Anatomie eines XLinks,
   - Arten von Links,
   - **Elemente und Attribute für XLink**

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Vokabular

- Jedes Element kann einen Link repräsentieren,
- XLinks werden nicht über spezielle Elemente definiert, sondern über Attribute,
- deren Namensraum ist http://www.w3.org/1999/xlink
- Jedes Element, dass einen Link repräsentiert, muss zwingend folgendes Attribut besitzen:

  `"xlink:type"`

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Vokabular – "xlink:type"

- "xlink:type" gibt die Art des XLink-Elementes an,
- Muss in jedem für XLink verwendeten Element zwingend vorhanden sein,

- Für einfache Links gilt:
  - Das Attribut "xlink:type" hat den Wert "simple"

- Für erweiterte Links gilt:
  - Das Attribut "xlink:type" hat den Wert "extended",
  - Kindelemente können den Link jetzt noch weiter spezifizieren, in diesem Fall besitzen diese Kindelemente das Attribut "xlink:type",
  - Folgende Möglichkeiten stehen zur Auswahl:
    - "locator",
    - "arc",
    - "resource",
    - "title"

adressdb.xml

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Vokabular

- Einfache XLinks können noch folgende Attribute besitzen:
- Das Attribut "`xlink:href`" ist im Gegensatz zu "`xlink:type`" optional,
- Weitere optionale Attribute für einfache Verweise sind:
  - "`xlink:role`" – beschreibt die Funktion des Verweises,
  - "`xlink:title`" – ein Titel für den Link,
  - "`xlink:show`" – Gibt an, wie die entfernte Ressource angezeigt werden soll:
    - `new:` öffnet ein z. B. neues Fenster,
    - `replace:` könnte den Inhalt des aktuellen Fensters ersetzen,
    - `embed:` eignet sich beispielsweise um den Inhalt der Ressource in das aktuelle Fenster einzufügen,
    - `undefined:` überlässt die Reaktion der Anwendung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Vokabular

- "`xlink:actuate`" – definiert, wie mit dem Link verfahren werden soll:
  - `onLoad:` lädt die verlinkte Ressource beim sofort beim Laden des Quelldokuments,
  - `onRequest:` lädt den Link erst wenn der Benutzer dazu den Befehl gibt,
  - `undefined:` überlässt es wieder der Anwendung

- Beispiel:

```
<bild xlink:href="bla.gif" xlink:show="embed"
xlink:actuate="onLoad" xlink:title="Ein Bild"/>
```

könnte das folgende, aus (X)HTML bekannte Konstrukt ersetzen:

```
<img src="bla.gif" alt="Ein Bild"/>
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Vokabular – Erweiterte Links

- "locator"-Typ:
  - Zeichnet die an einem erweiterten Link teilnehmenden **entfernten** Ressourcen aus,
  - Ist ein beliebiges Element mit dem Attributwert "locator" für das Attribut "type",
  - Besitzt zwingend das "href" Attribut,
  - Muss direktes Kindelement des erweiterten Linkelements sein,
  - Darf neben den XLink-Elementtypen weiteren beliebigen Inhalt haben

- Elemente vom Typ "locator" besitzen folgende Attribute:
  - "xlink:type" hat den Wert "locator",
  - "xlink:href" gibt den Ort der entfernten Ressource als URI an,
  - "xlink:role",
  - "xlink:title",
  - "xlink:label"

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Beispiele für erweiterte Links

- Definition einer entfernten Ressource:

adressdb.xml

```
<extendedlink xmlns:xlink="http://www.w3.org/1999/xlink"
   xlink:type="extended">
   <zvon_locator xlink:type="locator"
   xlink:href="http://www.zvon.org/xxl/XSLTreference/Outpu
   t/index.html">
    </zvon_locator>
</extendedlink>
```

Quelle: http://www.zvon.org/xxl/xlink/xlink_extend/OutputExamples

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Beispiele für erweiterte Links

- Definition einer lokalen Ressource:

```
<zvon_resource xmlns:xlink="http://www.w3.org/1999/xlink"
   xlink:type="resource">
    Anything here.
    <anytag>
      Any content.
    </anytag>
</zvon_resource>
```

- oder:

```
<zvon_resource xmlns:xlink="http://www.w3.org/1999/xlink"
   xlink:type="resource"/>
```

Quelle: http://www.zvon.org/xxl/xlink/xlink_extend/OutputExamples

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Beispiele für erweiterte Links

- Benutzung eines Title-Elementes:

```
<zvon_home xmlns:xlink="http://www.w3.org/1999/xlink"
   xlink:type="locator" xlink:title="Zvon">
   <zvon_title xlink:type="title" xml:lang="cs">
      Zvon
    </zvon_title>
   <zvon_title xlink:type="title" xml:lang="en">
      The Bell
    </zvon_title>
   <zvon_title xlink:type="title" xml:lang="de">
      Die Glocke
    </zvon_title>
</zvon_home>
```

Quelle: http://www.zvon.org/xxl/xlink/xlink_extend/OutputExamples

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Beispiele für erweiterte Links

- Benutzung eines Role-Elementes:

```
<container>
   <hero xmlns:xlink="http://www.w3.org/1999/xlink"
   xlink:type="locator"
   xlink:href="http://www.greece.antique/database/heracles.xml"
   xlink:role="http://www.greece.antique/hero"
   xlink:title="Heracles">
   </hero>
   <god xmlns:xlink="http://www.w3.org/1999/xlink"
   xlink:type="locator"
   xlink:href="http://www.greece.antique/database/zeus.xml"
   xlink:role="http://www.greece.antique/god"
   xlink:title="Zeus">
   </god>
   <woman xmlns:xlink="http://www.w3.org/1999/xlink"
   xlink:type="locator"
   xlink:href="http://www.greece.antique/database/alcmene.xml"
   xlink:role="http://www.greece.antique/woman"
   xlink:title="Alcmene">
   </woman>
</container>
```

Quelle: http://www.zvon.org/xxl/xlink/xlink_extend/OutputExamples

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Beispiele für erweiterte Links

- Benutzung eines Label-Elementes:

```
<woman
    xmlns:xlink="http://www.w3.org/1999/xlink"
    xlink:type="locator"
   xlink:href="http://www.greece.antique/database/alcmene.
   xml"
    xlink:role="http://www.greece.antique/woman"
    xlink:label="alcmene"
    xlink:title="Alcmene">
 </woman>
```

Quelle: http://www.zvon.org/xxl/xlink/xlink_extend/OutputExamples

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Beispiele für erweiterte Links

- Benutzung eines Arc-Elementes:

```
<mythology>
   <god
      xmlns:xlink="http://www.w3.org/1999/xlink"
      xlink:type="locator" ...
      xlink:label="zeus">
   </god>
   <hero
      xmlns:xlink="http://www.w3.org/1999/xlink"
      xlink:type="locator" ...
      xlink:label="heracles">
   </hero>
   <relation
      xmlns:xlink="http://www.w3.org/1999/xlink"
      xlink:type="arc"
      xlink:from="zeus"
      xlink:to="heracles">
   </relation>
 </mythology>
```

Quelle: http://www.zvon.org/xxl/xlink/xlink_extend/OutputExamples

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. XLink – Beispiele für erweiterte Links

- Benutzung eines Arc-Elementes mit Arcrole-Attribute:

```
<mythology
     xmlns:xlink="http://www.w3.org/1999/xlink"
   xlink:type="extended">
   <hero
      xmlns:xlink="http://www.w3.org/1999/xlink"
   xlink:type="locator" ...
      xlink:label="heracles">
   </hero>
   <god
      xmlns:xlink="http://www.w3.org/1999/xlink"
   xlink:type="locator" ...
      xlink:label="zeus">
   </god>
   <relation
      xmlns:xlink="http://www.w3.org/1999/xlink"
      xlink:type="arc" xlink:from="zeus" xlink:to="heracles"
   xlink:arcrole="http://www.greece.antique/relations/father">
   </relation>
 </mythology>
```

Quelle: http://www.zvon.org/xxl/xlink/xlink_extend/OutputExamples

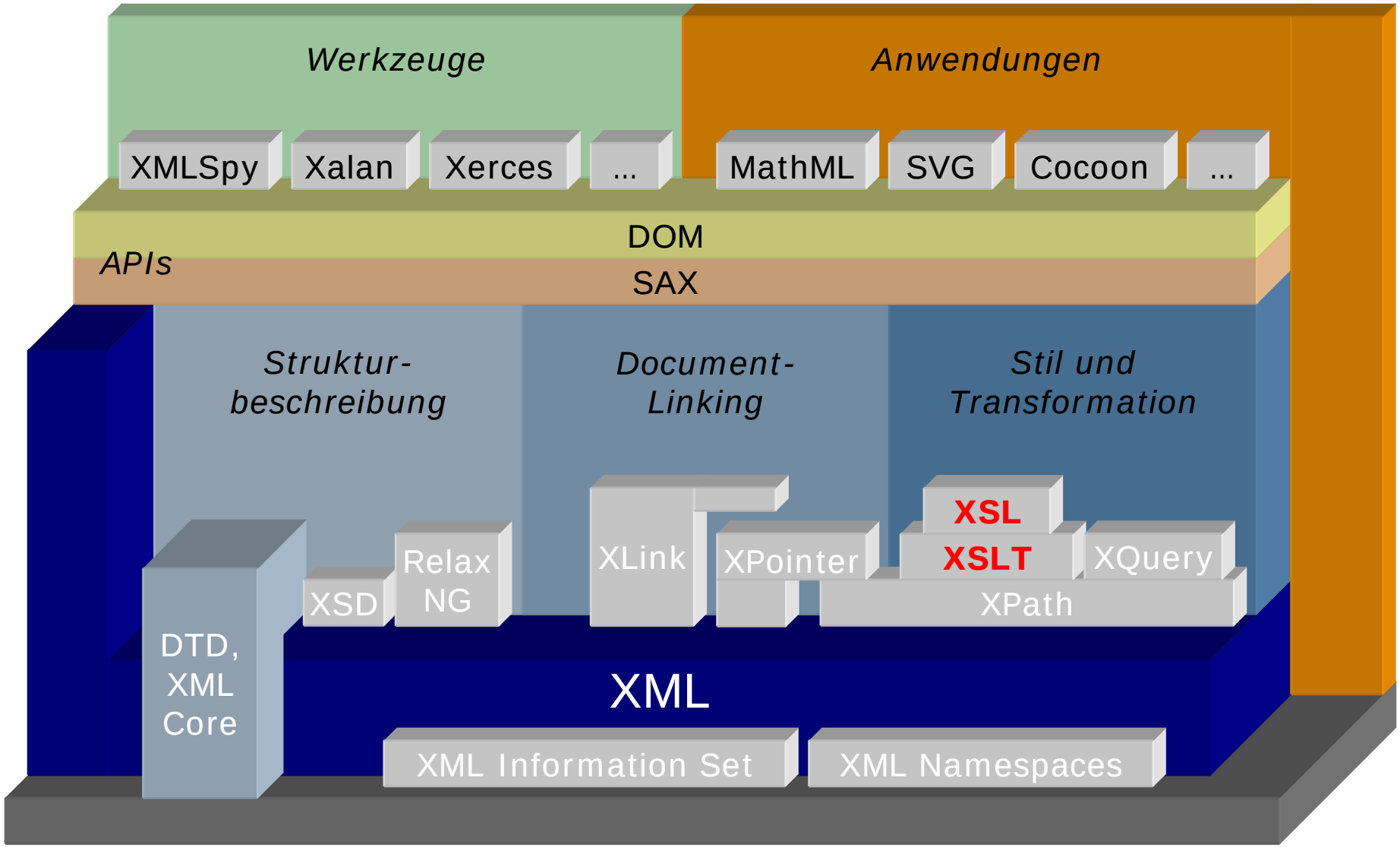
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Agenda

- XPointer
- XLink
- **XSLT**
- XSL-FO
- Fragen?
- Weiterer Ablauf

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Gliederung der Vorlesung



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Lernziele

- Die Motivation für XSLT und XSL kennen,
- Den Aufbau eines Stylesheets verstehen können,
- Wissen was Templates sind,
- Kontrollstrukturen von XSLT kennen,
- Einfache Stylesheets verstehen,
- Den Ablauf eines Publikationsprozess erläutern können
- die Aufgaben von XSL-FO verstehen

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

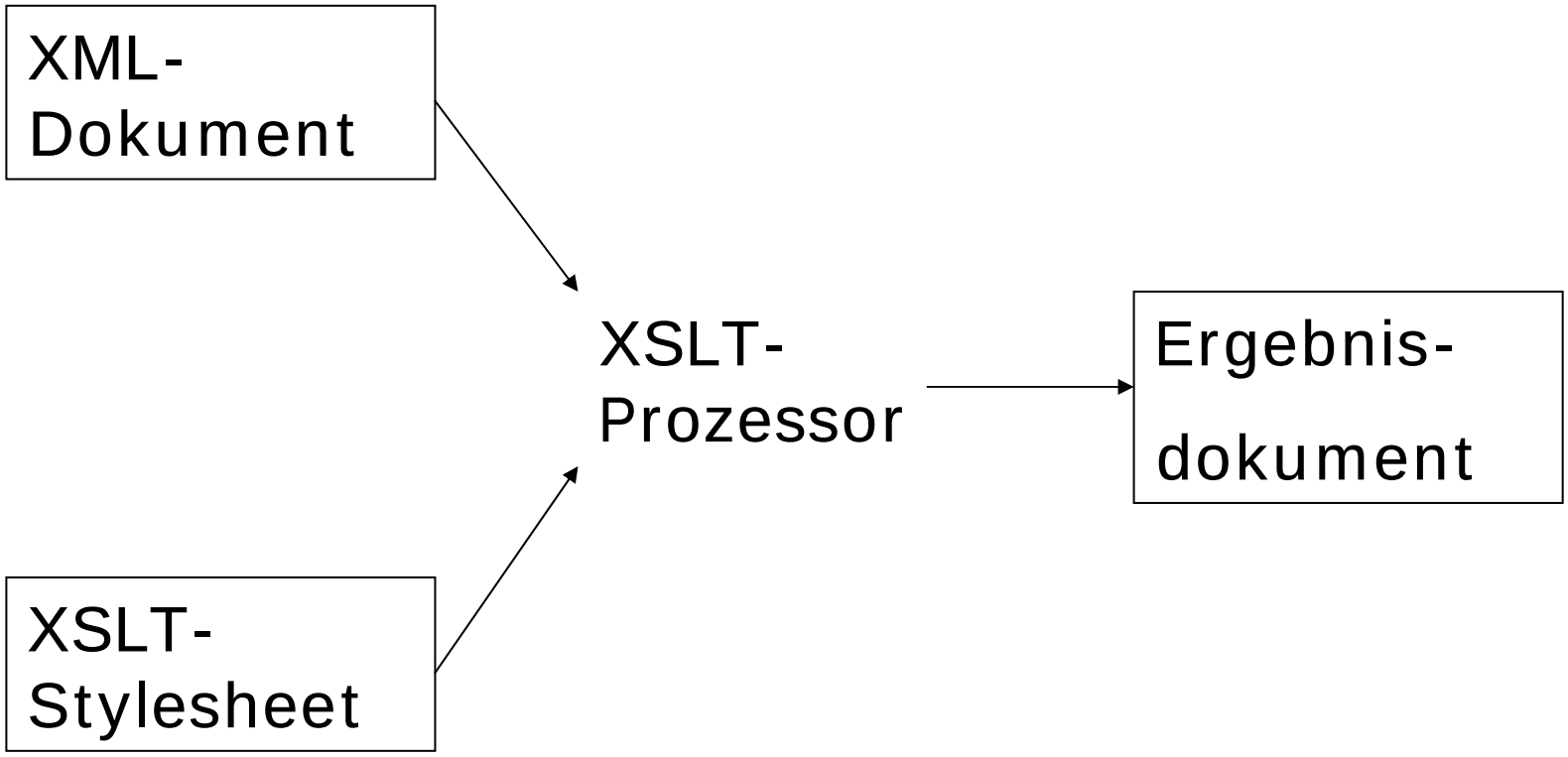
Gliederung

1. **Style & Transformations I – XSLT**
   1. **Motivation**
   2. Datenmodell
   3. Einordnung
   4. Stylesheet-Struktur
      - Templates,
      - Kontrollstrukturen,
      - Sortieren,
      - Anwendung von XPath-Ausdrücken
   5. XSLT 2.0

2. Style & Transformations II – XSL (FO)
   1. Motivation
   2. Einordnung
   3. Einführung
   4. Arten von Elementen
   5. Einsatzmöglichkeiten

1.1. Motivation

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme



Quelle: Vonhoegen

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Gliederung

1. **Style & Transformations I – XSLT**
   1. Motivation
   2. Datenmodell
   3. Einordnung
   4. **Stylesheet-Struktur**
      - Templates,
      - Kontrollstrukturen,
      - Sortieren,
      - Anwendung von XPath-Ausdrücken
   5. XSLT 2.0

2. Style & Transformations II – XSL (FO)
   1. Motivation
   2. Einordnung
   3. Einführung
   4. Arten von Elementen
   5. Einsatzmöglichkeiten

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Gliederung

1. **Style & Transformations I – XSLT**
   1. Motivation
   2. Datenmodell
   3. Einordnung
   4. Stylesheet-Struktur
      - **Templates**,
      - Kontrollstrukturen,
      - Sortieren,
      - Anwendung von XPath-Ausdrücken
   5. XSLT 2.0

2. Style & Transformations II – XSL (FO)
   1. Motivation
   2. Einordnung
   3. Einführung
   4. Arten von Elementen
   5. Einsatzmöglichkeiten

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1.4. Templates

Beispiel:

Wir wollen folgendes Dokument in ein HTML-Dokument transformieren.

Quelle:

helloworld/helloworld.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<gruss>
   Hello, World!
</gruss>
```

Ziel:

```
<body><h1>
   Hello, World!
</h1></body>
```

1.4. Templates

Das wird durch folgendes XSLT-Stylesheet erreicht:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
   <xsl:output method="html"/>
   <xsl:template match="/">
        <body>
              <h1>
                    <xsl:apply-templates select="gruss"/>
              </h1>
        </body>
   </xsl:template>
   <xsl:template match="gruss">
        <xsl:value-of select="."/>
   </xsl:template>
</xsl:stylesheet>
```

helloworld/helloworld.xsl

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1.4. Templates

Die genauere Steuerung des Quellbaumes kann folgendermaßen realisiert werden:

- Templates können auch benannt werden:
  ```
  <xsl:template match="vorname" name="vn_template">
  ```

  Das Pattern (Muster) als der Attributwert von `match` muss dabei einen XPath-Lokalisierungspfad enthalten, wobei nur die `child`- und `attribute`-Achsen und die Kurzformen `//` und `/` benutzt werden dürfen.

- Aufruf benannter Templates:
  ```
  <xsl:call-template name="vn_template"/>
  ```

  ```
  <xsl:apply-templates select="vorname"/>
  ```
  wählt alle Elementknoten zu mit `<vorname>` bezeichneten Elementen und führt dann die passenden Templates aus.

Parameter für Templates

Parameter können an ein Template übergeben werden, wenn es mit `<xsl:apply-templates>` oder `<xsl:call-template>` aufgerufen wird.

In beiden Fällen werden die Parameter als Kindelemente mit `<xsl:with-param>` festgelegt.

Der Parameterinhalt kann über das Attribut `select` oder als Inhalt des Elementes festgelegt werden.

Damit das aufgerufene Template die Parameter nutzen kann, müssen entsprechende `<xsl:param>`-Anweisungen als erste Kinder des Elementes `<xml:template>` eingefügt werden.
Default-Werte werden als Inhalt oder über das Attribut `select` angegeben.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1.4. Templates

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
   <xsl:output method="text"/>
<xsl:template match="/">
   <xsl:call-template name="summe">
        <xsl:with-param name="x1" select="30"/>
        <xsl:with-param name="x2">27</xsl:with-param>
   </xsl:call-template>
</xsl:template>
<xsl:template name="summe">
   <xsl:param name="x1"/>
   <xsl:param name="x2"/>
   <xsl:value-of select="$x1 + $x2"/>
</xsl:template>
</xsl:stylesheet>
```

xsl files/parameter.xslt

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Gliederung

1. **Style & Transformations I – XSLT**
   1. Motivation
   2. Datenmodell
   3. Einordnung
   4. Stylesheet-Struktur
      - Templates,
      - **Kontrollstrukturen**,
      - Sortieren,
      - Anwendung von XPath-Ausdrücken
   5. XSLT 2.0

2. Style & Transformations II – XSL (FO)
   1. Motivation
   2. Einordnung
   3. Einführung
   4. Arten von Elementen
   5. Einsatzmöglichkeiten

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1.4. Entscheidungen

Entscheidungen werden in XSLT durch die Anweisungen `<xsl:if>` und `<xsl:choose>` ermöglicht.

Beispiel: Auswahl eines bestimmten Buches aus einer Literaturliste

```
<xsl:template match="buch">
      <xsl:if test="titel='The XML Handbook'">
            ...
      </xsl:if>
</xsl:template>
```

Dabei können für das Attribut `test` XPath-Ausdrücke (Produktion `Expr`) verwendet werden. Der Inhalt von `xsl:if` ist wiederum ein Template.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1.4. Entscheidungen

Das Element `<xsl:choose>` kann ein oder mehrere Kindelemente `<xsl:when>` enthalten.

Zusätzlich kann mit dem Element `<xsl:otherwise>` ein else-Zweig angegeben werden.

```
...
<xsl:choose>
    <xsl:when test="note='1'">
        Sehr gut!
    </xsl:when>
    <xsl:otherwise>
        Naja...
    </xsl:otherwise>
</xsl:choose>
...
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1.4. Iteration

Iterationen werden durch die Anweisung `<xsl:for-each>` verwirklicht.

- Steuerung des Umfangs der Iteration durch das Attribut `select`
- durch dieses wird eine Menge von Knoten aus dem Quelldokument (ein XPath-`NodeSet`) ausgewählt
- Die als Kinder des Elementes aufgeführten Anweisungen (Template) werden dann nacheinander auf jeden Knoten dieser Knotenmenge angewendet

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1.4. Iteration

Beispiel (Ausgabe des Wertes des Kindelementes `titel` aller Kindelemente `buch` des Elementes `literaturliste`)**:**

```
...

<xsl:template match="literaturliste">
   <body>
      <xsl:for-each select="buch">
         <p>
            <xsl:value-of select="titel"/>
         </p>
      </xsl:for-each>
   </body>
</xsl:template>

...
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1.4. Variablen

Variablendeklarationen werden mit dem Element `<xsl:variable>` vorgenommen.

- Für globale Variablen als Kind von `<xsl:stylesheet>`

- für lokale Variablen innerhalb eines Templates

- mögliche Datentypen sind die, die ein XPath-Ausdruck zurückliefern kann: `string`, `number`, `boolean`, `node-set` und darüber hinaus Result Tree Fragments (Teil des Ausgabebaums)

- Wert kann nach der Deklaration nicht mehr geändert werden

- Variablen- und Parameternamen müssen eindeutig sein

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1.4. Variablen

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

   <xsl:variable name="c" select="'center'"/>

<xsl:template match="buch">
      <h3 align="{$c}">
         <xsl:value-of select="titel"/>
      </h3>
      <p align="{$c}">Verlag:
         <xsl:value-of select="verlag/name"/></p>
      <p align="{$c}">Ort:
         <xsl:value-of select="verlag/ort"/></p>
   </xsl:template>
</xsl:stylesheet>
```

xsl files/38.xslt

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Gliederung

1. **Style & Transformations I – XSLT**
   1. Motivation
   2. Datenmodell
   3. Einordnung
   4. Stylesheet-Struktur
      - Templates,
      - Kontrollstrukturen,
      - **Sortieren**,
      - Anwendung von XPath-Ausdrücken
   5. XSLT 2.0

2. Style & Transformations II – XSL (FO)
   1. Motivation
   2. Einordnung
   3. Einführung
   4. Arten von Elementen
   5. Einsatzmöglichkeiten

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1.4. Sortieren

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
   <xsl:output method="xml" version="1.0"
      encoding="UTF-8" indent="yes"/>

   <xsl:template match="/">
      <head>
         <title>Sortierte Literaturliste</title>
      </head>
      <body>
         <xsl:for-each select="/literaturliste/buch">
            <xsl:sort select="titel" data-type="text"
            order="ascending"/>
            <p><xsl:value-of select="titel"/></p>
         </xsl:for-each>
      </body>
   </xsl:template>
</xsl:stylesheet>
```

literaturliste/xsl/literaturliste_sortiert.xsl

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Gliederung

1. **Style & Transformations I – XSLT**
   1. Motivation
   2. Datenmodell
   3. Einordnung
   4. Stylesheet-Struktur
      - Templates,
      - Kontrollstrukturen,
      - Sortieren,
      - **Anwendung von XPath-Ausdrücken**
   5. XSLT 2.0

2. Style & Transformations II – XSL (FO)
   1. Motivation
   2. Einordnung
   3. Einführung
   4. Arten von Elementen
   5. Einsatzmöglichkeiten

1.4. Anwendung von XPath-Ausdrücken

Beispiel 1: Einfügen der Anzahl der Bücher der Literaturliste ins Zieldokument

```
...
<xsl:template match="/">
<html>
   <head>
      <title>Literaturliste (
         <xsl:value-of select="count(/literaturliste/buch)"/>
         Einträge)
      </title>
   </head>
...
```

Der Titel hat die Form *Literaturliste (x Einträge)*, mit x als Anzahl der eingetragenen Bücher.

1.4. Anwendung von XPath-Ausdrücken

Beispiel 2: Bücher mit mehreren Autoren

Ziel: Hat ein Buch genau einen Autor, so wird dessen Name ausgegeben. Hat es genau zwei Autoren, so werden deren Namen durch das Wort "und" verbunden.
Hat es mehr als zwei Autoren, so wird der Name des ersten eingetragenen Autors sowie die Ergänzung "u.a." ausgegeben.

Lösung: Bei der Verarbeitung für die Knoten `<buch>` wird die Anzahl der Kindelemente `<autor>` geprüft. Abhängig davon wird die Funktion `<xsl:apply-templates>` für diese mit dem Attribut `mode="1"`, `mode="2"` oder `mode="3"` (letzteres für mindestens 3 Autoren) aufgerufen. Dementsprechend werden dann die Templates für die Knoten `<autor>` und die drei Modi definiert.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1.4. Anwendung von XPath-Ausdrücken

```
...
<xsl:if test="count(./autor)&gt;2">
      <xsl:text>Autoren: </xsl:text>
      <xsl:apply-templates select="autor" mode="3"/>
</xsl:if>

<xsl:if test="count(./autor)=2">
      <xsl:text>Autoren: </xsl:text>
      <xsl:apply-templates select="autor" mode="2"/>
</xsl:if>

<xsl:if test="count(./autor)=1">
      <xs:text>Autor: </xs:text>
      <xsl:apply-templates select="autor" mode="1"/>
</xsl:if>
...
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1.4. Anwendung von XPath-Ausdrücken

Für genau zwei Autoren sieht das entsprechende Template z. B. wie folgt aus:

```
<xsl:template match="autor" mode="2">

   <xsl:if test="position()=last()">
       <xsl:text> und </xsl:text>
   </xsl:if>

   <xsl:value-of select="zusatz"/>
   <xsl:text> </xsl:text>
   <xsl:value-of select="name"/>
   <xsl:text>, </xsl:text>
   <xsl:apply-templates select="vorname"/>

</xsl:template>
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Gliederung

1. **Style & Transformations I – XSLT**
   1. Motivation
   2. Datenmodell
   3. Einordnung
   4. Stylesheet-Struktur
      - Templates,
      - Kontrollstrukturen,
      - Sortieren,
      - Anwendung von XPath-Ausdrücken
   5. **XSLT 2.0**

2. Style & Transformations II – XSL (FO)
   1. Motivation
   2. Einordnung
   3. Einführung
   4. Arten von Elementen
   5. Einsatzmöglichkeiten

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Erweiterungen in XSLT 2 (Auswahl)

- verwendet XPath-2.0-Ausdrücke
  - Datentypen, Sequenzen
  - Bedingungen und Schleifen
- Unterstützung von regulären Ausdrücken
  - xsl:analyze-string() zur Partitionierung von String in Substrings aufgrund regulärer Ausdrücke
  - andere: xsl:matching-substring(), xsl:non-matching-substring()
  - Funktionen in XPath 2: fn:match(), fn:replace(), fn:tokenize()
- Unterstützung von Gruppierungen
  - Beispiel siehe nächste Folien
- Unterstützung benutzerdefinierter Funktionen (xsl:function), die auch aus XPath-Ausdrücken heraus aufgerufen werden können
  - Beispiel siehe nächste Folien

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

XSLT2: Gruppierungen

```
<?xml version="1.0" encoding="UTF-8" ?>
<cars>
  <car make="Dodge" model="caravan" color="red" price="28000" />
  <car make="Ford" model="probe" color="blue" price="14000" />
  <car make="Ford" model="thunderbird" color="silver"
       price="45000"/>
  <car make="Ferrari" color="red" price="280000" />
  <car make="Dodge" model="caravan" color="green" price="28000" />
</cars>
```

Quelle: nach Kenton: What's new in XSLT2?

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

XSLT2: Gruppierungen

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes" />
  <xsl:template match="/">
    <colors>
      <xsl:for-each-group select="cars/car" group-by="@color">
        <color>
          <xsl:attribute name="name">
            <xsl:value-of select="@color" />
          </xsl:attribute>
          <xsl:element name="makes">
            <xsl:value-of select="current-group()/@make" separator="," />
          </xsl:element>
        </color>
      </xsl:for-each-group>
    </colors>
  </xsl:template>
</xsl:stylesheet>
```

xslt2/grouping/grouping.xslt

Quelle: nach Kenton: What's new in XSLT2?

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

XSLT2: Benutzerdefinierte Funktionen

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="2.0" xmlns:my="http://nowhere.org"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >
<xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes" />
<xsl:function name="my:vat">
  <xsl:param name="x" />
  <xsl:sequence select="$x * 0.16" />
</xsl:function>
<xsl:template match="/">
  <cars>
  <xsl:for-each select="cars/car">
    <car> <name>
      <xsl:value-of select="(@make, @model)" />
    </name>
    <vat><xsl:value-of select="my:vat(@price)" /></vat>
    </car>
  </xsl:for-each>
  </cars>
</xsl:template>
</xsl:stylesheet>
```

xslt2/user defined function/user_defined_function.xslt

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Erweiterungen in XSLT 2 (Auswahl)

- Unterstützung von Schemas
  - xsl:import-schema
- Unterstützung multipler Ausgaben (Dateien) in einem Stylesheet
  - Input
    - in XSLT 1.0 document()-Funktion zum Zugriff auf Dateien
    - in XSLT 2.0: input(), collection(), unparsed-text()
  - Output multipler Dokumente
    - xsl:result-document()
    -
```
<xsl:template match="/">
  ... primary output generation here ...
  <xsd:result-document href="result.wml">
  ... generation of WML output here...
  </xsd:result-document>
  <xsd:result-document href="result.voiceml">
  ... generation of VoiceML output here...
  </xsd:result-document>
</xsl:template>
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Erweiterungen in XSLT 2 (Auswahl)

- Verbesserte Internationalisierung
  - sprachabhängige Vergleiche
  - Formatierung von Zahlen und Daten
- Unterstützung von temporary trees als Ersatz für result tree fragments in XSLT 1.0
  - auf temporary trees können XPath-Ausdrücke operieren
  - Aufteilung komplexer Transformationen in mehrere Schritte
  - Beispiel siehe nächste Folien

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

XSLT2: Temporary Tree Fragments

```
<xsl:stylesheet version="2.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="xml" version="1.0" encoding="UTF-8"
  indent="yes" />
  <xsl:variable name="cars" select="/cars//car" />
  <xsl:template match="/">
    <cars>
      <expensive>
        <xsl:apply-templates select="$cars[@price > 50000]" />
      </expensive>
      <cheap>
        <xsl:apply-templates select="$cars[@price <= 50000]" />
      </cheap>
    </cars>
  </xsl:template>
  <xsl:template match="*">
    <xsl:copy-of select="." />
  </xsl:template>
</xsl:stylesheet>
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Agenda

- XPointer
- XLink
- XSLT
- **XSL-FO**
- Fragen?
- Weiterer Ablauf

Gliederung

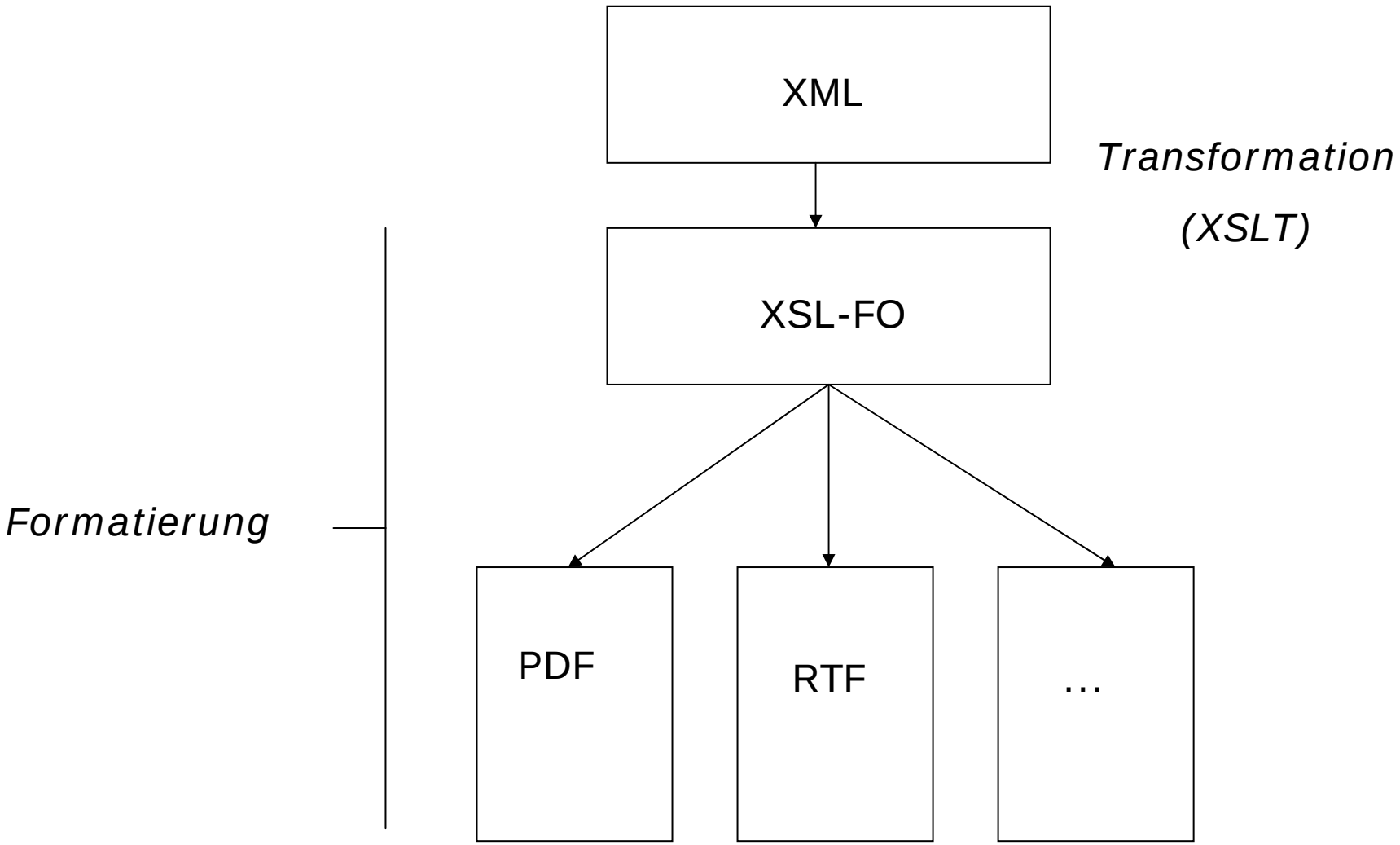
1. Style & Transformations I – XSLT
   1. Motivation
   2. Datenmodell
   3. Einordnung
   4. Stylesheet-Struktur
      - Templates,
      - Kontrollstrukturen,
      - Sortieren,
      - Anwendung von XPath-Ausdrücken
   5. XSLT 2.0

2. Style & Transformations II – XSL (FO)

   1. Motivation
   2. Einordnung
   3. Einführung
   4. Arten von Elementen
   5. Einsatzmöglichkeiten

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.1. Motivation



Gliederung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1. Style & Transformations I – XSLT
   1. Motivation
   2. Datenmodell
   3. Einordnung
   4. Stylesheet-Struktur
      - Templates,
      - Kontrollstrukturen,
      - Sortieren,
      - Anwendung von XPath-Ausdrücken
   5. XSLT 2.0

2. Style & Transformations II – XSL (FO)
   1. Motivation
   2. Einordnung
   3. Einführung
   4. Arten von Elementen
   5. Einsatzmöglichkeiten

Beispieldokument: Hello, World!

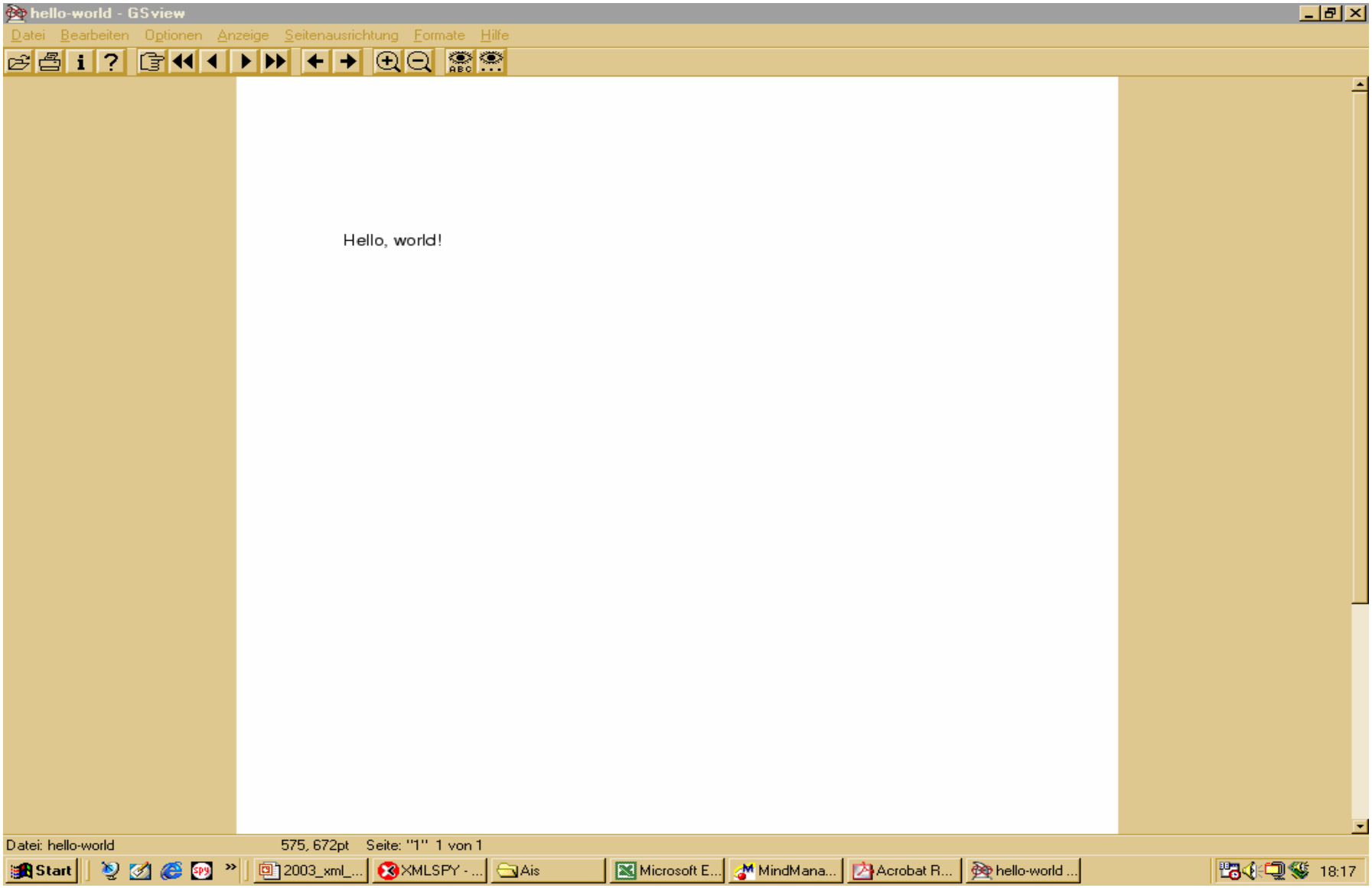
```
<?xml version="1.0" encoding="iso-8859-1"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <fo:layout-master-set>
       <fo:simple-page-master master-name="my-page">
             <fo:region-body margin="1in"/>
       </fo:simple-page-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="my-page">
       <fo:flow flow-name="xsl-region-body">
             <fo:block>Hello, world!</fo:block>
       </fo:flow>
   </fo:page-sequence>
</fo:root>
```

fo/helloworld_1.fo

Quelle: http://www.renderx.com/Tests/doc/html/tutorial.html

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.3. Einführung



Gliederung

1. Style & Transformations I – XSLT
   1. Motivation
   2. Datenmodell
   3. Einordnung
   4. Stylesheet-Struktur
      - Templates,
      - Kontrollstrukturen,
      - Sortieren,
      - Anwendung von XPath-Ausdrücken
   5. XSLT 2.0

2. Style & Transformations II – XSL (FO)
   1. Motivation
   2. Einordnung
   3. Einführung
   4. Arten von Elementen
   5. Einsatzmöglichkeiten

2.4. Elemente

Seitenaufbau

Als Kind des `<fo:simple-page-master>`-Elementes ist mindestens ein Bereich zur Aufnahme der Inhalte anzulegen:

Element	Repräsentiert den...
`<fo:region-body>`	zentralen Bereich
`<fo:region-before>`	Kopfbereich
`<fo:region-after>`	Fußbereich
`<fo:region-start>`	Bundbereich
`<fo:region-end>`	Außenbereich

- `<fo:flow>` und `<fo:static-content>` (beide zur Textausgabe; letzteres für fixierte Texte) können mit dem Attribut `flow-name` auf die angelegten Regionen zugreifen
- mögliche Werte: `xsl-region-body`, `xsl-region-before` usw.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.4. Elemente

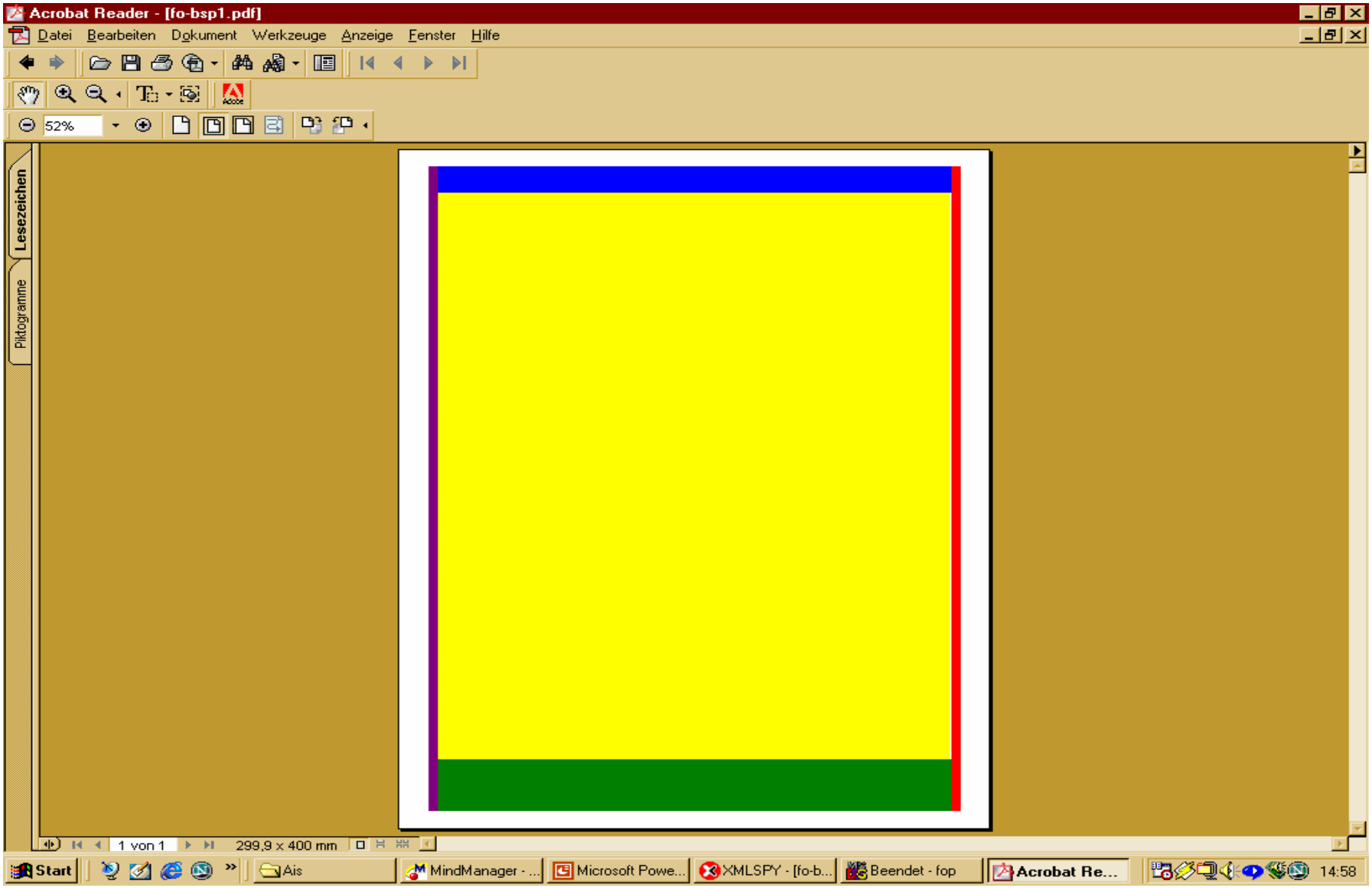
```
<?xml version="1.0" encoding="UTF-8"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <fo:layout-master-set>
     <fo:simple-page-master master-name="Seite"
        page-height="400mm" page-width="300mm"
         margin-top="10mm" margin-bottom="10mm"
         margin-right="15mm" margin-left="15mm">
        <fo:region-start extent="4mm" background-color="purple"/>
        <fo:region-before extent="15mm" background-color="blue"/>
        <fo:region-body margin-top="0mm" margin-bottom="0mm"
                        margin-left="5mm" margin-right="5mm"
                        background-color="yellow"/>
        <fo:region-after extent="30mm" background-color="green"/>

        <fo:region-end extent="4mm" background-color="red"/>
        </fo:simple-page-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="Seite">
        <fo:flow flow-name="xsl-region-body">
                                ...
```

fo/helloworld_2.fo

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.4. Elemente



Textformatierung

- `<fo:block>` definiert einen rechteckigen Anzeigebereich
- Über Attribute Absatzformatierung, z. B.:

Attribut	Bedeutung
`font-family`	Familie von Schriftarten; z. B. `"Helvetica"`
`font-size`	Schriftgröße; z. B. `"18pt"`
`background-color`	Hintergrundfarbe; z. B. `"lightblue"`

- Mit `<fo:inline>` können viele der möglichen Eigenschaften von Blöcken auch Zeichenfolgen innerhalb eines Blockes zugeordnet werden; z. B. erstes Zeichen eines Absatzes größer

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

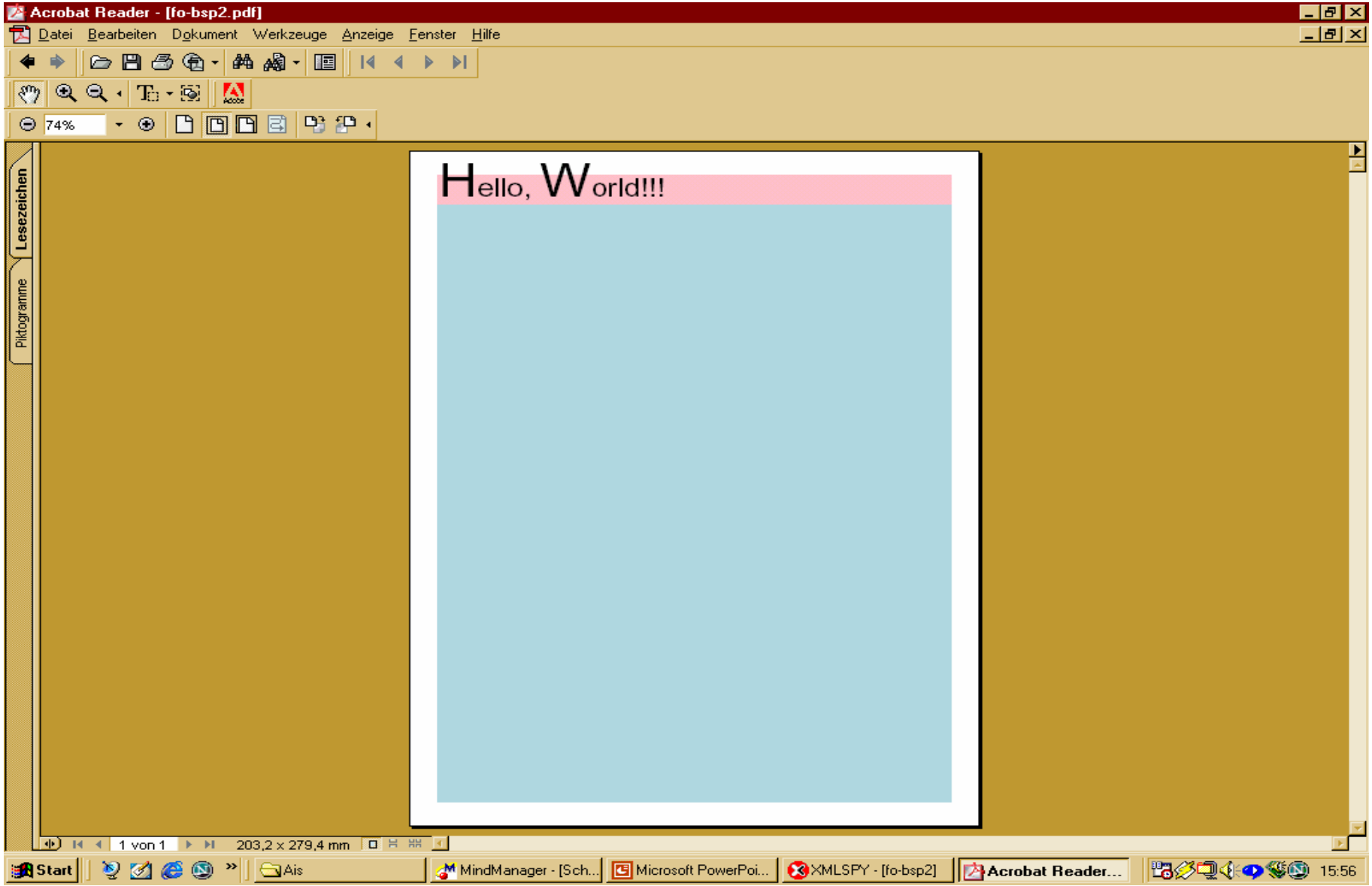
2.4. Elemente

```
<?xml version="1.0" encoding="UTF-8"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <fo:layout-master-set>
        <fo:simple-page-master master-name="Seite">
               <fo:region-body background-color="lightblue"
                                    margin="10mm"/>
        </fo:simple-page-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="Seite">
        <fo:flow flow-name="xsl-region-body">
               <fo:block background-color="pink" font-
                              family="Helvetica" font-
   size="30pt">
                        <fo:inline font-
   size="55pt">H</fo:inline>ello,
                        <fo:inline font-size="55pt">
   W</fo:inline>orld!!!
               </fo:block>
        </fo:flow>
   </fo:page-sequence>
</fo:root>
```

fo/helloworld_3.fo

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.4. Elemente



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.4. Elemente

Tabellen

Element	Bedeutung
`<fo:table>`	Repräsentiert die gesamte Tabelle
`<fo:table-body>`	Nimmt den Inhalt des Tabellenkörpers auf
`<fo:table-and-caption>`	Gruppiert die Elemente `<fo:table>` und `<fo:table-caption>`
`<fo:table-caption>`	Enthält Formatierungsobjekte, die die Überschrift der Tabelle enthalten; `<fo:table-and-caption>` muss verwendet werden
`<fo:table-cell>`	Platziert Inhalte in die Zellen
`<fo:table-column>`	Repräsentiert eine Spalte
`<fo:table-header>`	Nimmt den Inhalt des Tabellenkopfes auf
`<fo:table-row>`	Repräsentiert eine Zeile

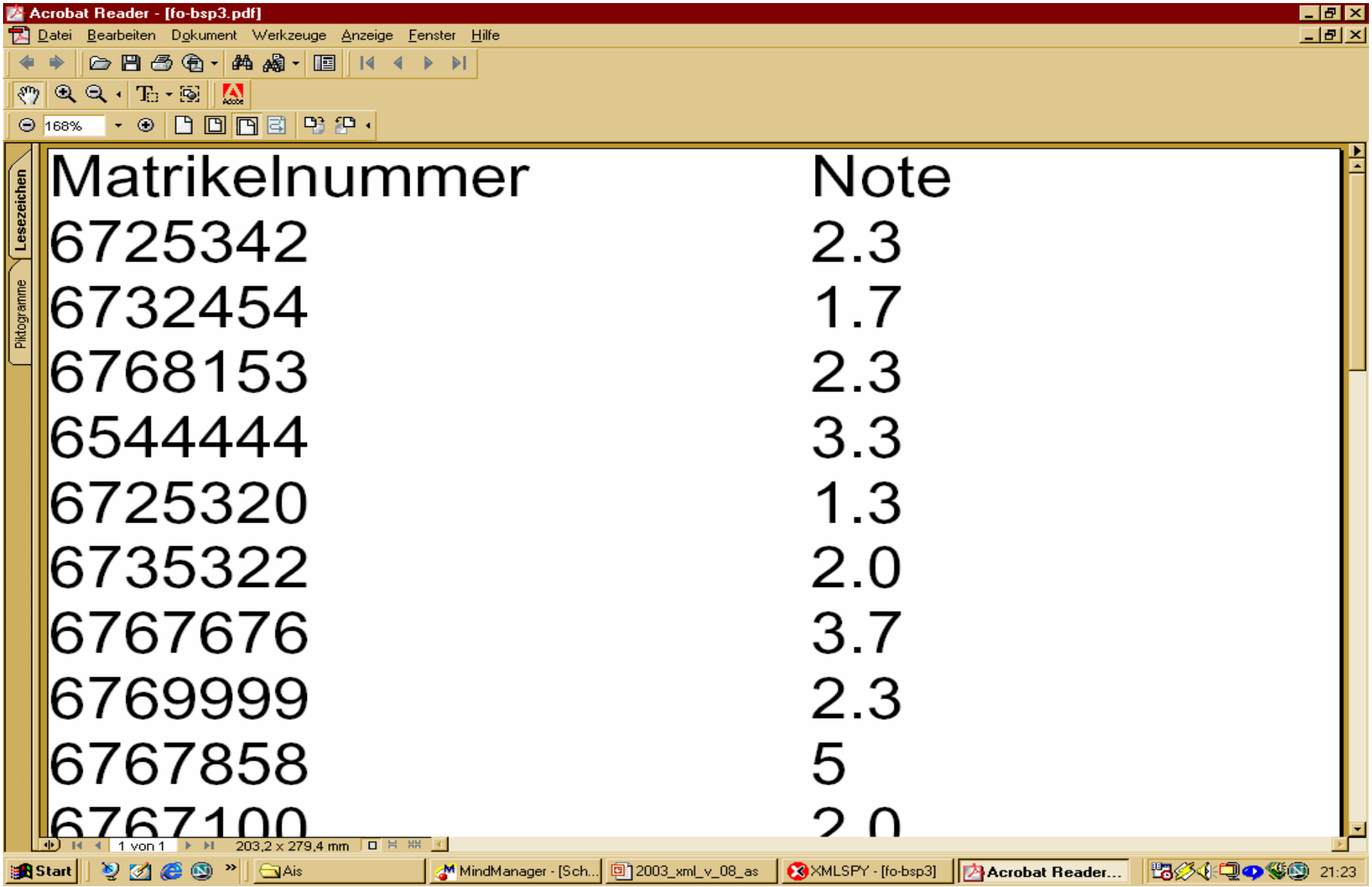
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.4. Elemente

```
...
<fo:flow flow-name="xsl-region-body">
  <fo:table font-size="30pt">
    <fo:table-column column-width="120mm" column-number="1"/>
    <fo:table-column column-width="40mm" column-number="2"/>
        <fo:table-body>
          <fo:table-row>
            <fo:table-cell column-number="1">
              <fo:block>Matrikelnummer</fo:block>
            </fo:table-cell>
            <fo:table-cell column-number="2">
              <fo:block>Note</fo:block>
            </fo:table-cell>
          </fo:table-row>
          <fo:table-row>
            <fo:table-cell column-number="1">
               <fo:block>6725342</fo:block>
            </fo:table-cell>
            <fo:table-cell column-number="2">
               <fo:block>2.3</fo:block>...
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.4. Elemente



2.4. Elemente

Listen

Element	Bedeutung
`<fo:list-block>`	Enthält die Liste
`<fo:list-item>`	Enthält das Label und den Körper eines Listeneintrags
`<fo:list-item-body>`	Enthält den eigentlichen Inhalt des Listeneintrags
`<fo:list-item-label>`	Enthält den Inhalt des Labels für den Listeneintrag

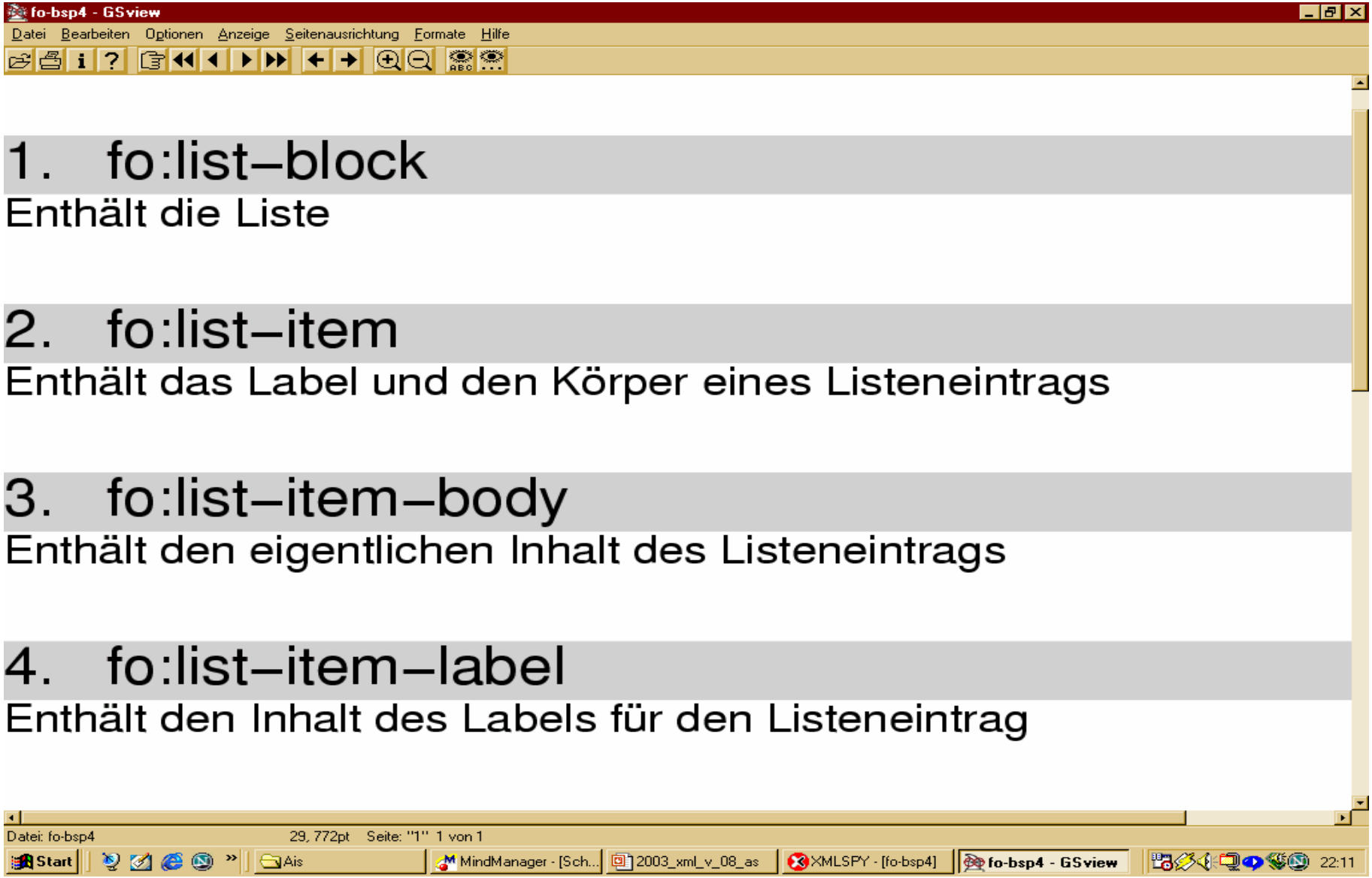
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.4. Elemente

```
...
<fo:page-sequence master-reference="Seite">
   <fo:flow flow-name="xsl-region-body">
        <fo:list-block>
          <fo:list-item font-size="24pt">
            <fo:list-item-label>
               <fo:block background-
                        color="lightgrey">1.</fo:block>
            </fo:list-item-label>
            <fo:list-item-body>
              <fo:block text-indent="40pt">
                fo:list-block</fo:block>
              <fo:block space-after="30pt" font-size="18pt">
                Enthält die Liste</fo:block>
            </fo:list-item-body>
          </fo:list-item>
          <fo:list-item font-size="24pt">
            <fo:list-item-label>
              <fo:block background-
                        color="lightgrey">2.</fo:block>...
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.4. Elemente



Gliederung

1. Style & Transformations I – XSLT
   1. Motivation
   2. Datenmodell
   3. Einordnung
   4. Stylesheet-Struktur
      - Templates,
      - Kontrollstrukturen,
      - Sortieren,
      - Anwendung von XPath-Ausdrücken
   5. XSLT 2.0

2. Style & Transformations II – XSL (FO)
   1. Motivation
   2. Einordnung
   3. Einführung
   4. Arten von Elementen
   5. Einsatzmöglichkeiten

Beispiel: Kompletter Publikationszyklus mit DocBook

- DocBook ist eine XML-Anwendung, die der Strukturierung von Artikeln, Büchern und Mengen selbiger dient
  - http://www.docbook.org/
- Zahlreiche XSLT-Stylesheets für die Transformation in z. B.:
  - FO
  - HTML
  - Windows-Hilfe
  - Java-Hilfe

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.5. Anwendungen: Einfaches DocBook-File

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE book PUBLIC "-//OASIS//DTD DocBook XML V4.2//EN"
                "http://www.oasis-
   open.org/docbook/xml/4.2/docbookx.dtd">
<?xmlspysps
   http://www.altova.com/sps/Template/Publishing/docbook.sps?>
<book lang="de">
      <title>XSL</title>
      <chapter>
            <title>Ein kurzes Buch</title>
            <sect1>
                  <title>Problemstellung</title>
                  <para> Es muss ein DocBook-Beispiel erstellt
                     <indexterm>
                           <primary>DocBook</primary>
                     </indexterm> werden, das auf eine Folie
   passt.          </para>
            </sect1>
      </chapter>
</book>
```

docbook/docbook_sample.xml

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.5. Anwendungen

Das fertige DocBook-Dokument könnte z. B. so aussehen:

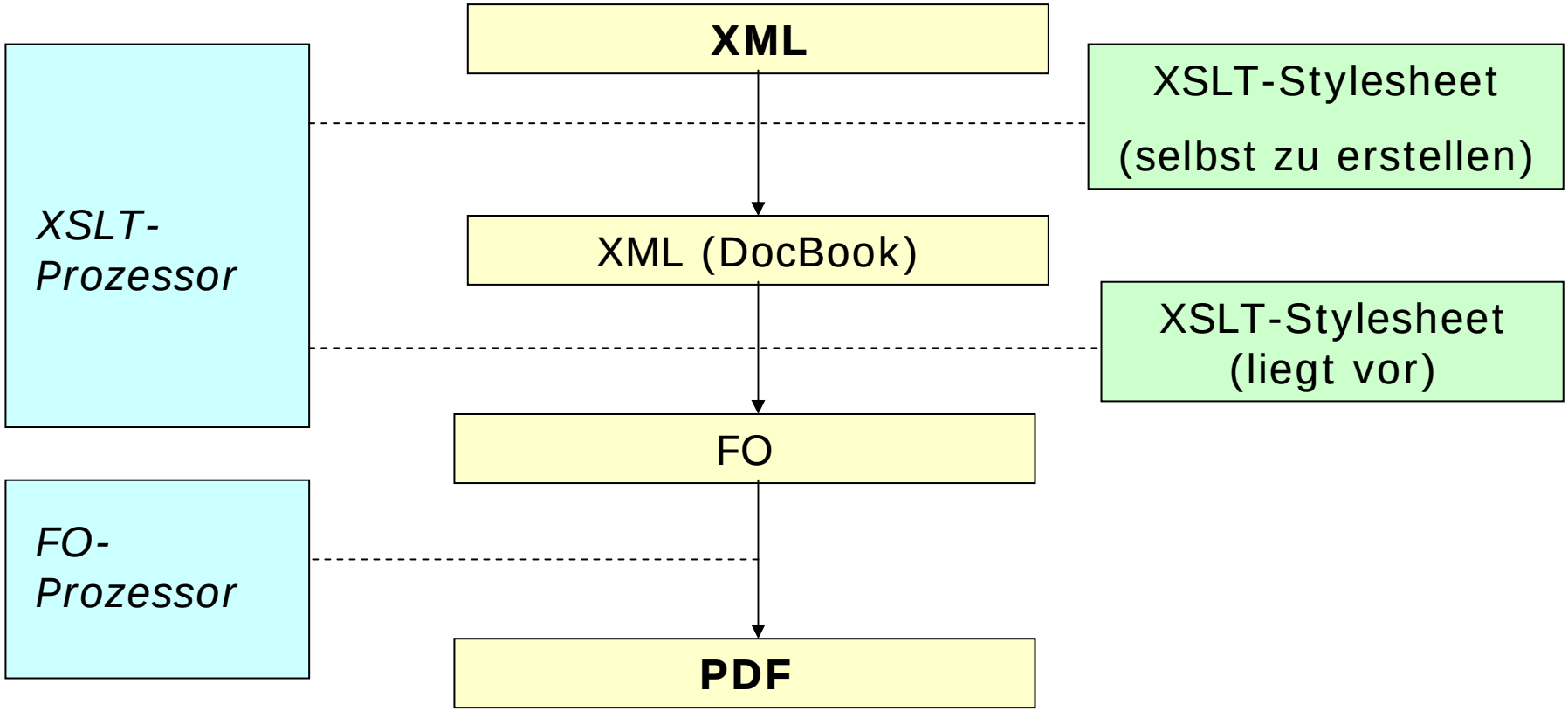
```
<?xml version="1.0" encoding="UTF-8"?>
<article lang="de"><articleinfo>
    <title>XML-Tools</title>
    <subtitle>Kleine Übersicht</subtitle>
    <abstract>
      <para>
         Zum Arbeiten mit XML und den verwandten Techniken
  stehen mittlerweile recht viele Tools zur Verfügung. Nicht
  alle implementieren bereits den gesamten Standard, und
  unterschiedliche Tools können unterschiedliche Ergebnisse
  liefern. Hier soll kurz eine Übersicht über kostenlose Tools
  und auch solche, von denen zumindest eine Demoversion
  kostenlos verfügbar ist, gegeben werden.
    </para>
  </abstract>
  </articleinfo>
  <section>
    <title>XMLSpy</title>
    <para>
    XMLSpy ist in erster Linie ein XML-Editor. Die Software
    enthält...
```

docbook/docbook_xmltools.xml

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.5. Anwendungen

In unserem Beispiel liegen die zu veröffentlichenden Daten in Form eines oder mehrerer XML-Dokumente vor. Sie sollen ins PDF-Format übertragen werden.



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.5. Anwendungen

Auszug aus dem XSLT-Stylesheet DocBook ➔ FO:

```
...
<xsl:template match="/">
 <xsl:variable name="document.element" select="*[1]"/>
  <xsl:if test="not(contains( $root.elements, concat(' ', local-
        name($document.element), ' ')))">
  <xsl:message terminate="yes">
        ERROR: Document root element for FO output
        must be one of the following elements:
        <xsl:value-of select="$root.elements"/>
  </xsl:message>
  </xsl:if>
  <xsl:call-template name="root.messages"/>
  <xsl:variable name="title">
    <xsl:choose>
      <xsl:when test="$document.element/title[1]">
        <xsl:value-of select="$document.element/title[1]"/>
      </xsl:when>
      <xsl:otherwise>
        [could not find document title]
      </xsl:otherwise>
                ...
```

demo/docbook/docbook/fo/docbook.xsl

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.5. Anwendungen

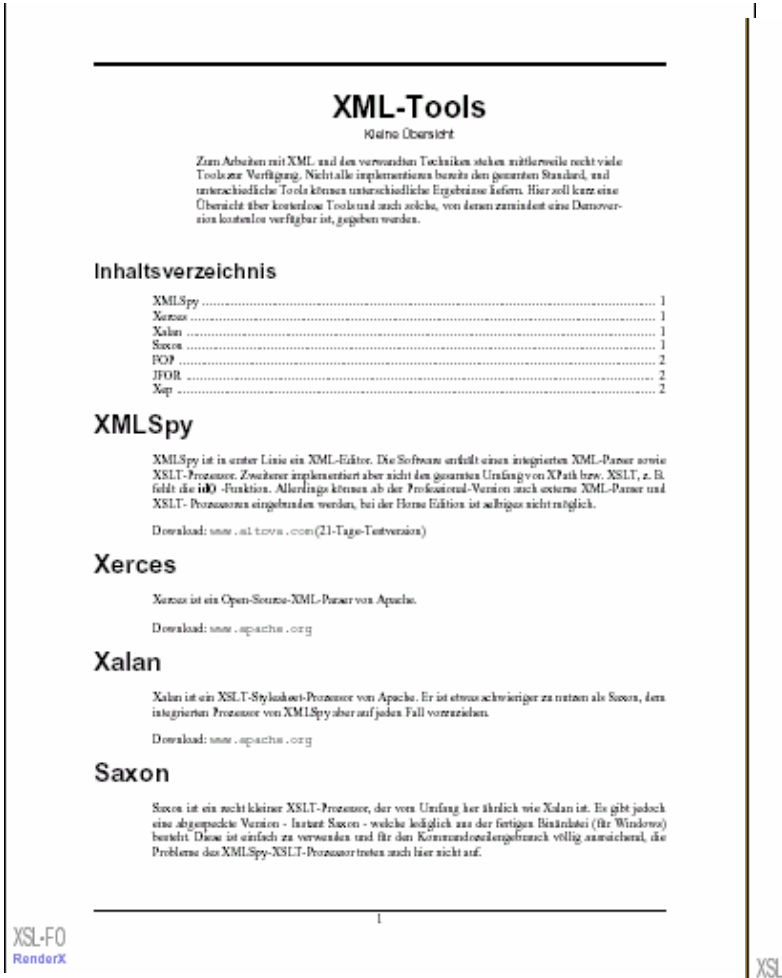
Mit Saxon als XSLT-Prozessor und dem Stylesheet für die Transformation von DocBook in FO entsteht:

<?xml version="1.0" encoding="utf-8"?><fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format" font-family="serif,Symbol,ZapfDingbats" font-size="10pt" text-align="justify" line-height="normal" font-selection-strategy="character-by-character" language="de"><fo:layout-master-set><fo:simple-page-master master-name="blank" page-width="8.5in" page-height="11in" margin-top="0.5in" margin-bottom="0.5in" margin-left="1in - -4pc" margin-right="1in"><fo:region-body display-align="center" margin-bottom="0.5in" margin-top="0.5in" region-name="blank-body"/><fo:region-before region-name="xsl-region-before-blank" extent="0.4in" display-align="before"/><fo:region-after region-name="xsl-region-after-blank" extent="0.4in" display-align="after"/></fo:simple-page-master><fo:simple-page-master master-name="titlepage-first" page-width="8.5in" page-height="11in" margin-top="0.5in" margin-bottom="0.5in" margin-left="1in - -4pc" margin-right="1in"><fo:region-body margin-bottom="0.5in" margin-top="0.5in" column-gap="12pt" column-count="1"/><fo:region-before region-name="xsl-region-before-first" extent="0.4in" display-align="before"/><fo:region-after region-name="xsl-region-after-first" extent="0.4in" display-align="after"/></fo:simple-page-master><fo:simple-page-master master-name="titlepage-odd" page-width="8.5in" page-height="11in" margin-top="0.5in" margin-bottom="0.5in" margin-left="1in - -4pc" margin-right="1in"><fo:region-body margin-bottom="0.5in" margin-top="0.5in" column-gap="12pt" column-count="1"/><fo:region-before region-name="xsl-region-before-odd" extent="0.4in" display-align="before"/><fo:region-after region-name="xsl-region-after-odd" extent="0.4in" display-align="after"/></fo:simple-page-master><fo:simple-page-master master-name="titlepage-even" page-width="8.5in" page-height="11in" margin-top="0.5in" margin-bottom="0.5in" margin-right="1in" margin-left="1in - -4pc"><fo:region-body margin-bottom="0.5in" margin-top="0.5in" column-gap="12pt" column-count="1"/><fo:region-before region-name="xsl-region-before-even" extent="0.4in" display-align="before"/><fo:region-after region-name="xsl-region-after-even" extent="0.4in" display-align="after"/></fo:simple-page-master><fo:simple-page-master master-name="lot-first" page-width="8.5in" page-height="11in" margin-top="0.5in" margin-bottom="0.5in" margin-left="1in - -4pc" margin-right="1in"><fo:region-body margin-bottom="0.5in" margin-top="0.5in" column-gap="12pt" column-count="1"/><fo:region-before region-name="xsl-region-before-first" extent="0.4in" display-align="before"/><fo:region-after region-name="xsl-region-after-first" extent="0.4in" display-align="after"/></fo:simple-page-master><fo:simple-page-master master-name="lot-odd" page-width="8.5in" page-height="11in" margin-top="0.5in" margin-bottom="0.5in" margin-left="1in - -4pc" margin-right="1in"><fo:region-body margin-bottom="0.5in" margin-top="0.5in" column-gap="12pt" column-count="1"/><fo:region-before region-name="xsl-region-before-odd" extent="0.4in" display-align="before"/><fo:region-after region-name="xsl-region-after-odd" extent="0.4in" display-align="after"/></fo:simple-page-master><fo:simple-page-master master-name="lot-even" page-width="8.5in" page-height="11in" margin-top="0.5in" margin-bottom="0.5in" margin-right="1in" margin-left="1in - -4pc"><fo:region-body margin-bottom="0.5in" margin-top="0.5in" column-gap="12pt" column-count="1"/><fo:region-before region-name="xsl-region-before-even" extent="0.4in" display-align="before"/><fo:region-after region-name="xsl-region-after-even" extent="0.4in" display-align="after"/></fo:simple-page-master><fo:simple-page-master master-name="front-first" page-width="8.5in" page-height="11in" margin-top="0.5in" margin-bottom="0.5in" margin-left="1in - -4pc" margin-right="1in"><fo:region-body margin-bottom="0.5in" margin-top="0.5in" column-gap="12pt" column-count="1"/><fo:region-before region-name="xsl-region-before-first" extent="0.4in" display-align="before"/><fo:region-after region-name="xsl-region-after-first" extent="0.4in" display-align="after"/></fo:simple-page-master><fo:simple-page-master master-name="front-odd" page-width="8.5in" page-height="11in" margin-top="0.5in" margin-bottom="0.5in" margin-left="1in - -4pc" margin-right="1in"><fo:region-body margin-bottom="0.5in" margin-top="0.5in" column-gap="12pt" column-count="1"/><fo:region-before region-name="xsl-region-before-odd" extent="0.4in" display-align="before"/><fo:region-after region-name="xsl-region-after-odd" extent="0.4in" display-align="after"/></fo:simple-page-master><fo:simple-page-master master-name="front-even" page-width="8.5in" page-height="11in" margin-top="0.5in" margin-bottom="0.5in" margin-right="1in" margin-left="1in - -4pc"><fo:region-body margin-bottom="0.5in" margin-top="0.5in" column-gap="12pt" column-count="1"/><fo:region-before region-name="xsl-region-before-even" extent="0.4in" display-align="before"/><fo:region-after region-name="xsl-region-after-even" extent="0.4in" display-align="after"/>...

docbook/docbook_xmltools.fo

2.5. Anwendungen

Mit Xep als FO-Prozessor entsteht daraus:



XML-Tools

Kleine Übersicht

Zum Arbeiten mit XML und den verwandten Techniken stehen mittlerweile recht viele Tools zur Verfügung. Nicht alle implementieren bereits den gesamten Standard, und unterschiedliche Tools können unterschiedliche Ergebnisse liefern. Hier soll kurz eine Übersicht über kostenlose Tools und auch solche, von denen zumindest eine Demoversion kostenlos verfügbar ist, gegeben werden.

Inhaltsverzeichnis

XMLSpy

XMLSpy ist in erster Linie ein XML-Editor. Die Software enthält einen integrierten XML-Parser sowie XSLT-Prozessor. Zweiterer implementiert aber nicht den gesamten Umfang von XPath bzw. XSLT, z. B. fehlt die **id()** -Funktion. Allerdings können ab der Professional-Version auch externe XML-Parser und XSLT- Prozessoren eingebunden werden, bei der Home Edition ist selbiges nicht möglich.

Download: www.altova.com(21-Tage-Testversion)

Xerces

Xerces ist ein Open-Source-XML-Parser von Apache.

Download: www.apache.org

Xalan

Xalan ist ein XSLT-Stylesheet-Prozessor von Apache. Er ist etwas schwieriger zu nutzen als Saxon, dem integrierten Prozessor von XMLSpy aber auf jeden Fall vorzuziehen.

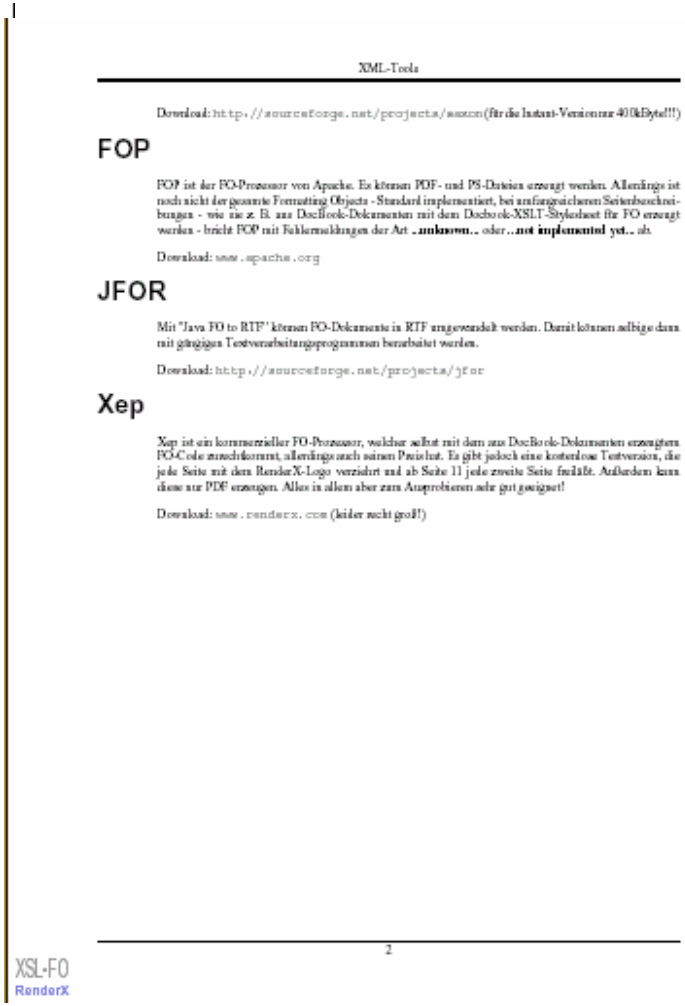
Download: www.apache.org

Saxon

Saxon ist ein recht kleiner XSLT-Prozessor, der vom Umfang her ähnlich wie Xalan ist. Es gibt jedoch eine abgespeckte Version - Instant Saxon - welche lediglich aus der fertigen Binärdatei (für Windows) besteht. Diese ist einfach zu verwenden und für den Kommandozeilengebrauch völlig ausreichend, die Probleme des XMLSpy-XSLT-Prozessor treten auch hier nicht auf.

1

XSL-FO
RenderX



XML-Tools

Download: http://sourceforge.net/projects/saxon (für die Instant-Version nur 400kByte!!!)

FOP

FOP ist der FO-Prozessor von Apache. Es können PDF- und PS-Dateien erzeugt werden. Allerdings ist noch nicht der gesamte Formatting Objects - Standard implementiert, bei unfangreichen Seitenbeschreibungen - wie sie z. B. aus DocBook-Dokumenten mit dem DocBook-XSLT-Stylesheet für FO erzeugt werden - bricht FOP mit Fehlermeldungen der Art **..unknown..** oder **..not implemtid yet..** ab.

Download: www.apache.org

JFOR

Mit "Java FO to RTF" können FO-Dokumente in RTF umgewandelt werden. Damit können selbige dann mit gängigen Textverarbeitungsprogrammen bearbeitet werden.

Download: http://sourceforge.net/projects/jfor

Xep

Xep ist ein kommerzieller FO-Prozessor, welcher selbst mit dem aus DocBook-Dokumenten erzeugten FO-Code einwandfrei umgeht, allerdings auch seinen Preis hat. Es gibt jedoch eine kostenlose Testversion, die jede Seite mit dem RenderX-Logo verziert und ab Seite 11 jede zweite Seite freilässt. Außerdem kann diese nur PDF erzeugen. Alles in allem aber zum Ausprobieren sehr gut geeignet!

Download: www.renderx.com (leider nicht groß!)

2

XSL-FO
RenderX

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Agenda

- XPointer
- XLink
- XSLT
- XSL-FO
- **Fragen?**
- Weiterer Ablauf

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Agenda

- XPointer
- XLink
- XSLT
- XSL-FO
- Fragen?
- **Weiterer Ablauf**

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Weiterer Ablauf

- Selbststudium des Materials
  - Beispiele eigenständig nachvollziehen
  - Unklarheiten sind die Basis für die Diskussion in den Präsenzveranstaltungen
- Präsenzveranstaltungen (Termine vorläufig, montags, 09:15–10:45 Uhr, Johannisgasse 26, R 1-22)
  1. 2008-05-19: Einführung/Strukturbeschreibung (SK)
  2. 2008-06-09: Adressierung, Abfrage und Speicherung (MT)
  3. 2008-06-23: Document-Linking/Transformation und Präsentation (MT)
  4. 2008-07-14: APIs und Werkzeuge, Semantic Web (MG, TR)
  5. 2008-07-31f: Modulprüfung BIS
- Aktuelle Informationen siehe http://bis.informatik.uni-leipzig.de/