

XSL/XSLT

4th Arcanum

Jägerstr. 24
D-65187 Wiesbaden
info@4th-arcanum.de

Kontakt:

Andreas Mertens

Mobil +49 178-88 55 686
Fax +49 178 99 88 55 686
Email andreas.mertens@4th-arcanum.de

Diese Unterlage ist
urheberrechtsgeschützt.

© 2003–2008 Andreas Mertens

XSLT

Die Informationen in diesem Produkt werden ohne Rücksicht auf einen eventuellen Patentschutz veröffentlicht. Warennamen werden ohne Gewährleistung der freien Verwendbarkeit genutzt. Bei der Zusammenstellung von Texten und Abbildungen wurde mit größter Sorgfalt vorgegangen. Trotzdem können Fehler nicht vollständig ausgeschlossen werden. Der Autor kann für fehlerhafte Angaben und deren Folgen weder eine juristische Verantwortung noch irgendeine Haftung übernehmen. Für Verbesserungsvorschläge und Hinweise sind wir dankbar.

Alle Rechte vorbehalten, auch die der fotomechanischen Wiedergabe und der Speicherung in elektronischen Medien. Die gewerbliche Nutzung der in diesem Produkt gezeigten Modelle und Arbeiten ist nicht zulässig.

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Kurs-Überblick

- Dieser Kurs ist **stark** praxisorientiert
- Aus diesem Grund werden die meisten Inhalte auf Basis von konkreten Beispielen demonstriert
- Wir springen deshalb immer wieder zwischen reinen XSLT-Themen und XPath-Themen hin und her
- In der Praxis sind die Themen ebenfalls ineinander verwoben

überblick und architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Kurs-Überblick

- Zunächst werden wir uns einen Überblick verschaffen, was der Unterschied zwischen einem XML-Parser und einem XML-Prozessor ist
- Wir werden uns anschauen, wie der XML-Parser und der XML-Prozessor in die „PHP/Apache-Architektur“ eingebettet ist
- Für die Übungen werden wir uns ansehen, wie man den Prozessor von der Shell aufruft (den sonst PHP aufruft)

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Kurs-Überblick

- Wir werden uns den Unterschied zwischen XPath und XSLT genau anschauen
- Dies wird an einem Beispiel demonstriert

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Kurs-Überblick

- Wir werden uns dann anschauen, wie man mit den sogenannten XPath-Axes in einem Dokumentenbaum „navigiert“.
- Dies werden wir an einer Aufgabe nachvollziehen und vertiefen

überblick und
architektur

xslt und xpath

xpath axes

**xslt kontroll-
strukturen**

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Kurs-Überblick

- Anschließend schauen wir uns eine wichtige XSLT-Kontrollstruktur an: `<xsl:if>`
- Wir werden angelehnt an die Dialog-DTD bzw. XML-Struktur mit einem Stylesheet das `<Ident>`-Tag untersuchen und das Attribut in dem `<Ident filename=" ">`-Tag auf `.eps`-Dateien oder `.tif`-Dateien untersuchen:
 - `<Ident filename="grafik1.eps"/>`
 - `<Ident filename="grafik1.tif"/>`

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Kurs-Überblick

- Wir werden und anschauen, wie man einem XML-Prozessor besondere Parameter mitgeben kann, um sein Verhalten von außen zu steuern
- Wir schauen uns dann Parameterübergaben aus dem PHP-Kontext an den Prozessor an

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Kurs-Überblick

- Mittels spezieller XPath-Funktionen, mit den wir Strings manipulieren können, wandeln wir ein deutsches Datumsformat in ein Englisches

```
DateDemo_1.xml  
<GermanDate>15.07.2003</GermanDate>
```



```
DateDemo_Result1.xml  
<EnglishDate>2003-07-15</EnglishDate>
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

**weitere xslt-
funktionen**

xpath-
funktionen II

xslt-funktionen

Kurs-Überblick

- Mit besonderen XSLT-Funktionen, werden wir die Möglichkeit ein Datum zu verwandeln modularisieren
- Dadurch können wir uns eine eigene kleine „Bibliothek“ mit Funktionen bauen, auf die wir immer wieder zurückgreifen
- Das Rad muss dann nicht immer neu erfunden werden => Wir sparen Zeit (!)

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

**xpath-
funktionen II**

xslt-funktionen

Kurs-Überblick

- Mit weiteren XPath-Funktionen werden wir rechnen
- Wir berechnen den Durchschnittspreis einer Buchsammlung

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Kurs-Überblick

- Das reicht noch nicht?
- Wir werden lernen, wie man mit einem Stylesheet mehrere XML-Dokumente gleichzeitig verarbeitet, um z.B. mehrere XML-Quellen zusammenzuführen
- Die Krönung kommt zum Schluß: Wie ruft man JavaScript direkt aus Stylesheets auf?

überblick und architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

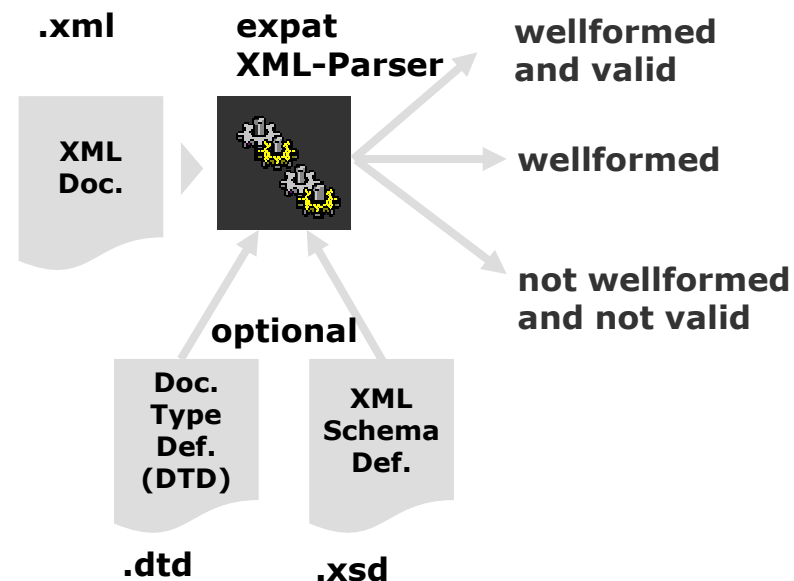
xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Wiederholung



XML=eXtended Markup Language
XSD = XML Schema Definition
DTD = Document Type Definition
XSL = XML Stylesheet Language
PDF = Portable Document Format
HTML = Hypertext Markup Language

überblick und architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

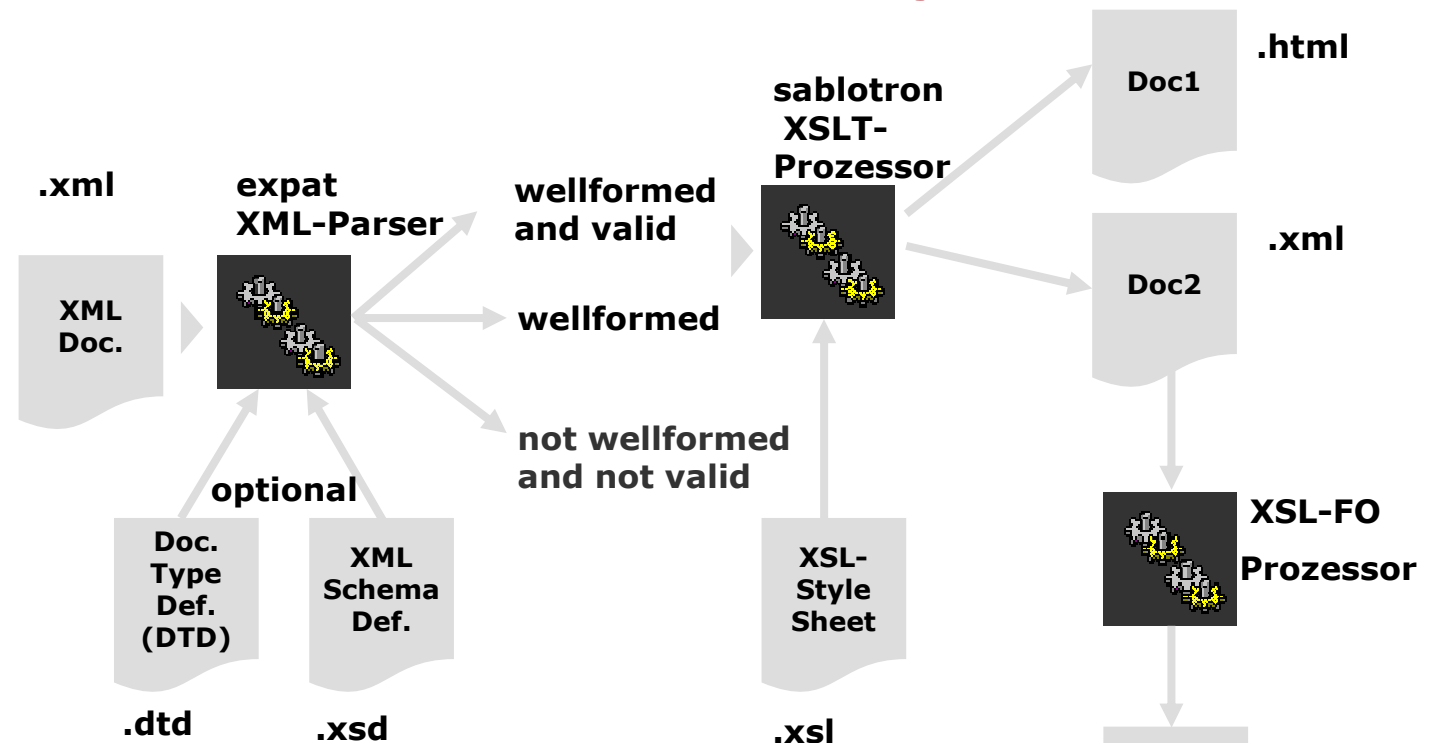
xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Wiederholung



XML=eXtended Markup Language
XSD = XML Schema Definition
DTD = Document Type Definition
XSL = XML Stylesheet Language
PDF = Portable Document Format
HTML = Hypertext Markup Language

überblick und architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

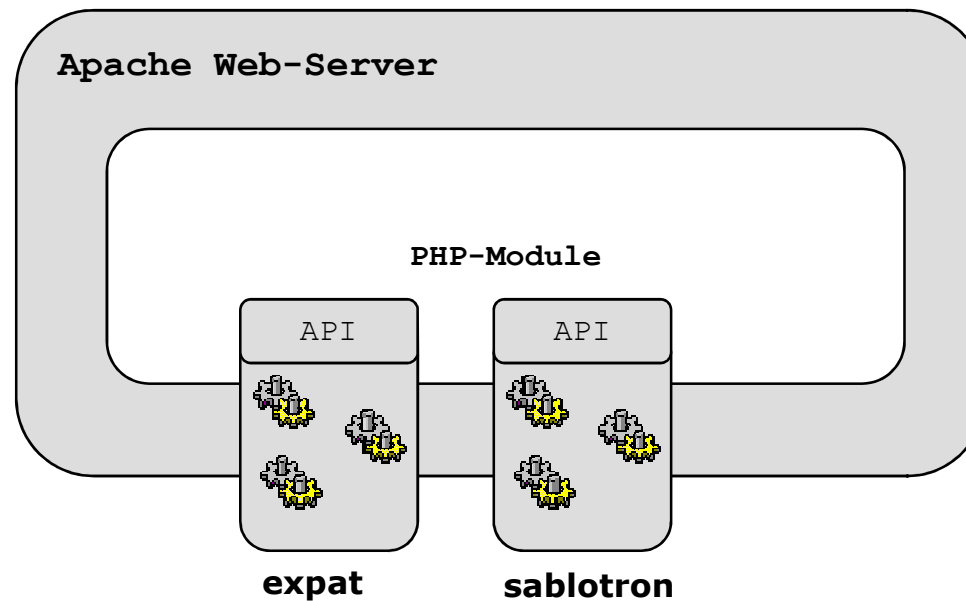
xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Apache Architektur



überblick und architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Sablotron

- www.gingerall.org
- Aktuelle Version 0.98
- Benötigt XMLParser expat von James Clark
- ECMAScript (JavaScript)-Extensions vorhanden
- Keine Java-Extension

überblick und architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Sablotron von der Shell

- Der XSLT-Prozessor lässt sich auch von der Shell aus ausführen, um XSLT-Stylesheets ohne PHP zu testen:

sabcmd <stylesheet> <infile> <outfile>

```
linuxsamsung:/srv/www/htdocs # sabcmd --help
Usage:
  sabcmd [options] <stylesheet> [<input> [<output>]] [assignments]
Options:
  --base=NAME, -b          set the hard base URI to NAME
  --debug-options          display information on debugging options
  --help, -?              display this help message
  --log-file=NAME, -L     set the log file, turn logging on
  --measure, -m           measure the processing time
  --version, -v           display version information
Defaults:
  <input> = stdin, <output> = stdout
  logging off, no hard base URI
Notes:
  - assignments define named buffers (name=value)
    and top-level params ($name=value).
  - to specify value in an option, use -b=NAME or -b NAME
    (correspondingly for the equivalent long options)
linuxsamsung:/srv/www/htdocs # _
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

XSLT und XPath

1. XSLT-Elemente sind XML-Tags wie

```
<xsl:template match="XPath-Ausdruck" .../>
```

```
<xsl:apply-templates select="..." .../>
```

```
<xsl:stylesheet .../>
```

2. XPath wird von XSLT-Elementen benutzt durch die **match** bzw. **select**-Attribute

überblick und
architektur

xslt und xpath

xml-struktur / attribute
xslt & xpath
template / apply-template
stylesheet einbinden
//-Operator

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp.: XSLT_XPathDemo_1 (1)

1. Gegeben ist folgendes XML-File

XSLT_XPathDemo_1.xml

```
...  
<DialogStory DialogVersion="3.4">  
  <StoryHead>  
    <Ident/>  
    <DocAttr>  
      <Level>  
        <Publisher Id="1" CustId="ST" Name="Schwäbisches Tagblatt"/>  
      </Level>  
      <InternetAttr/>  
      <PubDate PubDateBegin="10.05.2003" PubDateEnd="11.05.2003"/>  
      <Editor Id="98" Name="mat"/>  
    </DocAttr>  
  </StoryHead>  
  <StoryTechData>  
    <Preview>ENTRINGEN. Es ist die letzte Gemeinderatssitzung</Preview>  
  </StoryTechData>  
</DialogStory>
```

überblick und
architektur

xslt und xpath

xml-struktur / attribute
xslt & xpath
template / apply-template
stylesheet einbinden
//-Operator

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp.: XSLT_XPathDemo_1 (1)

1. Gegeben ist folgendes XML-File

```
XSLT_XPathDemo_1.xml
...
<DialogStory DialogVersion="3.4">
  <StoryHead>
    <Ident/>
    <DocAttr>
      <Level>
        <Publisher Id="1" CustId="ST" Name="Schwäbisches Tagblatt"/>
      </Level>
      <InternetAttr/>
      <PubDate PubDateBegin="10.05.2003" PubDateEnd="11.05.2003"/>
      <Editor Id="98" Name="mat"/>
    </DocAttr>
  </StoryHead>
  <StoryTechData>
    <Preview>ENTRINGEN. Es ist die letzte Gemeinderatssitzung</Preview>
  </StoryTechData>
</DialogStory>
```

Der Inhalt des Attributes **PubDateBegin** des XML-Elementes **PubDate** soll selektiert werden unter Verwendung von XSLT und XPath

überblick und
architektur

xslt und xpath

xml-struktur / attribute
xslt & xpath
template / apply-template
stylesheet einbinden
//-Operator

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp.: XSLT_XPathDemo_1 (2)

Das zugehörige Stylesheet, XSLT und XPath:

```
XSLT_XPathDemo_1.xsl  
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"  
  version="1.0">  
  <xsl:output method="html"/>  
  <xsl:template match="/">  
    <xsl:apply-templates select="DialogStory/StoryHead/DocAttr/PubDate"/>  
  </xsl:template>  
  
  <xsl:template match="PubDate">  
    <xsl:value-of select="./@PubDateBegin"/>  
  </xsl:template>  
  
</xsl:stylesheet>
```

XSLT: Vorlage erzeugen

XPath: Wurzel selektieren

XPath: XML Element PubDate selektieren und Attribut PubDateBegin ausgeben

überblick und
architektur

xslt und xpath

xml-struktur / attribute

xslt & xpath

template / apply-template

stylesheet einbinden

//-Operator

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp.: XSLT_XPathDemo_1 (3)

Wie funktioniert es?

XSLT_XPathDemo_1.xsl

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  version="1.0">
  <xsl:output method="html"/>

  <xsl:template match="/">
    <xsl:apply-templates
      select="DialogStory/StoryHead/
        DocAttr/PubDate"/>
  </xsl:template>

  <xsl:template match="PubDate">
    <xsl:value-of select="./@PubDateBegin"/>
  </xsl:template>

</xsl:stylesheet>
```

**XSLT: Im Wurzel-Kontext (/)
wird das PubDate-Template
mit <xsl:apply-template/>
aufgerufen**

überblick und
architektur

xslt und xpath

xml-struktur / attribute

xslt & xpath

template / apply-template

stylesheet einbinden

//-Operator

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp.: XSLT_XPathDemo_1 (4)

Das Resultat ist lediglich das Datum:

XSLT_XPathDemo_1.out
10.05.2003

Das Stylesheet kann entweder im XML-File
angegeben werden oder manuell dem XSLT-
Prozessor als Parameter übergeben werden.

überblick und
architektur

xslt und xpath

xml-struktur / attribute
xslt & xpath
template / apply-template
stylesheet einbinden
//-Operator

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp.: XSLT_XPathDemo_1 (5)

Stylesheet im XML-File:

```
XSLT_XPathDemo_1.xml
<?xml-stylesheet type="text/xsl" href="XSLT_XPathDemo_1.xsl"?>
.
.
```

Oder über Parameter beim Prozessoraufruf:

```
sabcmd <stylesheet> <infile> <outfile>
```

```
sabcmd XSLT_XPathDemo_1.xsl
      XSLT_XPathDemo_1.xml
      XSLT_XPathDemo_1.out
```

überblick und
architektur

xslt und xpath

xml-struktur / attribute
xslt & xpath
template / apply-template
stylesheet einbinden
//-Operator

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp.: XSLT_XPathDemo_2 (1)

- Folgendes Beispiel liest das Attribut **PubDateBegin** aus, erzeugt diesmal allerdings eine HTML-Seite mit einer Tabelle. Zusätzlich werden alle XML-Elemente `<Editor/>` rekursiv ausgelesen.

überblick und
architektur

xslt und xpath

xml-struktur / attribute
xslt & xpath
template / apply-template
stylesheet einbinden
//-Operator

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp.: XSLT_XPathDemo_2 (2)

Gegeben ist folgendes XML-File:

XSLT_XPathDemo_2.xml

```
<DialogStory DialogVersion="3.4">
  <StoryHead>
    <Ident/>
    <DocAttr>
      <Level>
        <Publisher Id="1" CustId="ST" Name="Schwäbisches Tagblatt"/>
      </Level>
      <InternetAttr/>
      <PubDate PubDateBegin="10.05.2003" PubDateEnd="11.05.2003"/>
      <Editor Id="98" Name="mat"/>
    </DocAttr>
    <Editor Id="98" Name="mat2"/>
  </StoryHead>
  <StoryTechData>
    <Preview>ENTRINGEN. Es ist die letzte Gemeinderatssitzung</Preview>
    <Editor Id="98" Name="mat3"/>
  </StoryTechData>
</DialogStory>
```

überblick und
architektur

xslt und xpath

xml-struktur / attribute
xslt & xpath
template / apply-template
stylesheet einbinden
//-Operator

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

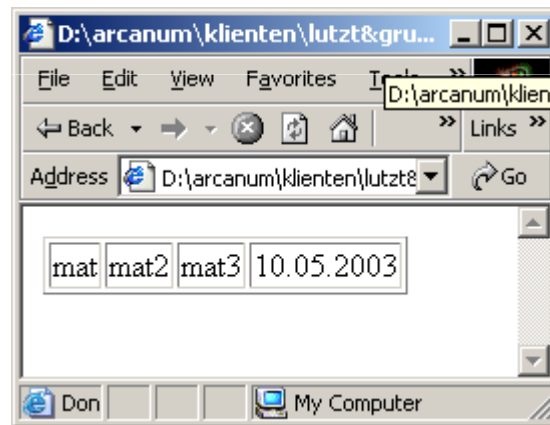
weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp.: XSLT_XPathDemo_2 (3)

- Das gewünschte Ergebnis soll wie folgt aussehen:



überblick und
architektur

xslt und xpath

xml-struktur / attribute
xslt & xpath
template / apply-template
stylesheet einbinden
//-Operator

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp.: XSLT_XPathDemo_2 (4)

Das zugehörige Stylesheet:

XSLT_XPathDemo_2.xsl

```
<xsl:stylesheet xmlns:xsl=http://www.w3.org/1999/XSL/Transform
version="1.0">
<xsl:output method="html"/>
<xsl:template match="/">
<html><body><table border="1"><tr>
  <xsl:apply-templates select="//Editor"/>
  <xsl:apply-templates select="DialogStory/StoryHead/DocAttr/PubDate"/>
</tr></table></body></html>
</xsl:template>

<xsl:template match="PubDate">
  <td><xsl:value-of select="./@PubDateBegin"/></td>
</xsl:template>

<xsl:template match="Editor">
  <td><xsl:value-of select="./@Name"/></td>
</xsl:template>
</xsl:stylesheet>
```

**XPath: rekursiver
Operator //**

XPath: Attribute selektieren

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

XPath Axes (Achsen)

Innerhalb eines XPath-Ausdruckes gibt es die sog. XPath-Achsen, um Attribute und Elemente zu adressieren.

Insgesamt gibt es 13 verschiedene Achsen

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

XPath – Axes

- | | |
|------------------------------|------------------------------|
| 1. child | 9. preceding-sibling |
| 2. parent | 10. following-sibling |
| 3. self | 11. preceding |
| 4. attribut | 12. following |
| 5. ancestor | 13. namespace |
| 6. ancestor-or-self | |
| 7. descendant | |
| 8. descendant-or-self | |

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

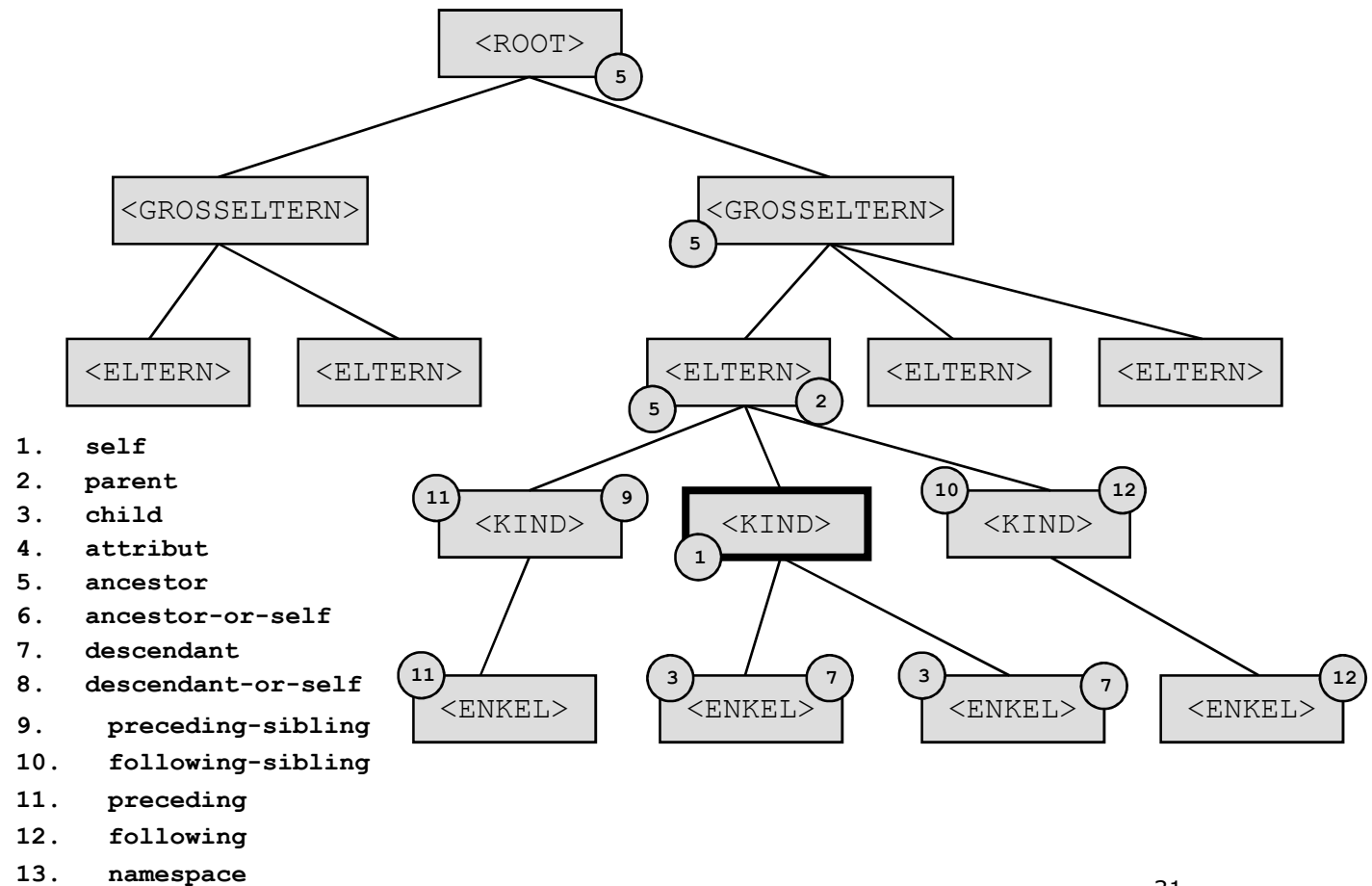
xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

XPath – Axes



überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp. zu XPath-Achsen:

Folgendes Stylesheet selektiert das Datum aus dem Beispiel (**XSLT_XPathDemo_1.xml**) mit XPath-Achsen auf verschiedene Arten

XSLT_XPathDemo_1XPathAxes.xsl

```
...  
<xsl:template match="PubDate">  
    <xsl:value-of select="self::*/@PubDateBegin"/>  
  
    <xsl:value-of select="attribute::PubDateBegin"/>  
  
    <xsl:value-of select="./@PubDateBegin"/>  
</xsl:template>  
...
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp. Kopieren eines Teilbaumes:

Der Teilbaum unter dem Element `DocAttr` soll
kopiert werden.

XSLT_XPathDemo_Child.xml

```
<DialogStory DialogVersion="3.4">
  <StoryHead>
    <Ident/>
    <DocAttr>
      <Level>
        <Publisher Id="1" CustId="ST" Name="Schwäbisches Tagblatt"/>
      </Level>
      <InternetAttr/>
      <PubDate PubDateBegin="10.05.2003" PubDateEnd="11.05.2003"/>
      <Editor Id="98" Name="mat"/>
    </DocAttr>
    ...
  </StoryHead>
</DialogStory>
```

Kopieren

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp. Kopieren eines Teilbaumes:

Folgendes Stylesheet kopiert den Teilbaum
DocAttr in den Ergebnisbaum

XSLT_XPathDemo_Child.xsl

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"  
version="1.0">  
  
<xsl:output method="xml"/>  
<xsl:template match="/DialogStory/StoryHead/DocAttr/child::*">  
    <xsl:copy-of select="."/>  
</xsl:template>  
  
</xsl:stylesheet>
```

**XSLT: Teilbaum in
Ergebnisbaum mit
<xsl:copy-of/>
kopieren**

**XPath: selektieren mit
XPath-child-Achse**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp. Kopieren eines Teilbaumes:

Oder folgende Alternativen:

XSLT_XPathDemo_Child.xsl

```
<xsl:output method="xml"/>
<xsl:template match="/DialogStory/StoryHead/DocAttr/child::*">
    <xsl:copy-of select="."/>
</xsl:template>
```

XSLT_XPathDemo_Child2.xsl

```
<xsl:template match="/DialogStory/StoryHead/child::DocAttr/child::*">
    <xsl:copy-of select="."/>
</xsl:template>
```

XSLT_XPathDemo_Child3.xsl

```
<xsl:template match="/DialogStory/StoryHead">
    <xsl:copy-of select="./child::DocAttr/child::*">
</xsl:template>
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

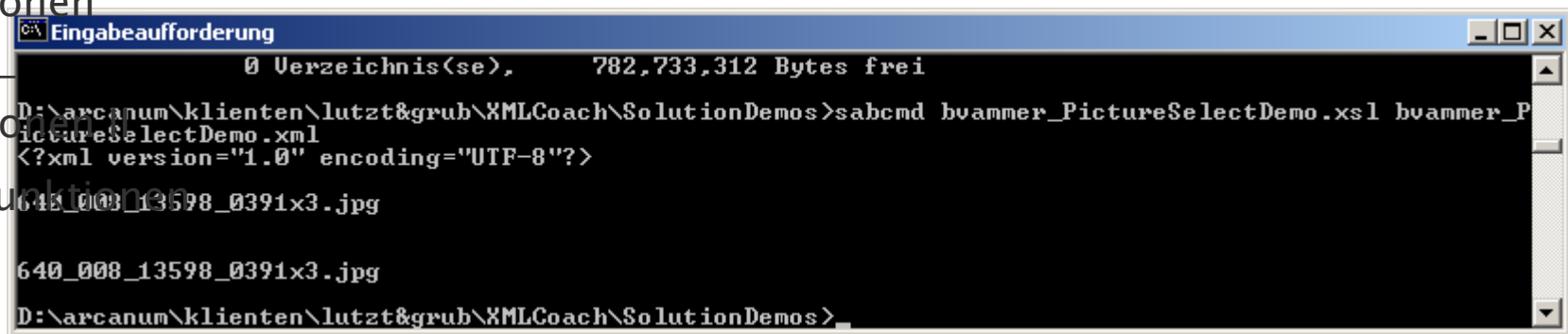
xpath-

funktionen II

xslt-funktionen

Bsp. Kontrollstrukturen

Folgendes Beispiel gibt unter Verwendung des `<xsl:if>`-Elements in dem Element `<Ident>` der Dialog-Dokumentenstruktur alle .jpg-Dateien aus. Das Ergebniss aus der Datei `bvammer.xml` sieht Wie folgt aus:



```
Eingabeaufforderung
C:\  Verzeichnis(se), 782,733,312 Bytes frei
D:\arcanum\klienten\lutz&grub\XMLCoach\SolutionDemos>sabcmd bvammer_PictureSelectDemo.xsl bvammer_PictureSelectDemo.xml
<?xml version="1.0" encoding="UTF-8"?>
640_008_13598_0391x3.jpg
640_008_13598_0391x3.jpg
D:\arcanum\klienten\lutz&grub\XMLCoach\SolutionDemos>
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp. Kontrollstrukturen

Stringvergleiche mit `<xsl:if/>`:

Bvammer_PictureSelectDemo.xsl

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
<xsl:output method="xml"/>
<xsl:template match="/">
    <xsl:apply-templates select="//Ident"/>
</xsl:template>

<xsl:template match="Ident">
    <xsl:variable name="extension">.jpg</xsl:variable>
    <xsl:variable name="filename" select="./@PathName"/>
    <xsl:text>&#x0A;</xsl:text>
    <xsl:if test="substring($filename, string-length($filename)
- string-length($extension)+1) = $extension">
        <xsl:value-of select="./@PathName"/>
    </xsl:if>
</xsl:template>
</xsl:stylesheet>
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp. Kontrollstrukturen

Stringvergleiche mit `<xsl:if/>`:

Bvammer_PictureSelectDemo.xsl

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
<xsl:output method="xml"/>
<xsl:template match="/">
    <xsl:apply-templates select="//Ident"/>
</xsl:template>

<xsl:template match="Ident">
    <xsl:variable name="extension">.jpg</xsl:variable>
    <xsl:variable name="filename" select="$file" />
    <xsl:text>&#x0A;</xsl:text>
    <xsl:if test="substring($filename, string-length($extension)) = $extension">
        <xsl:value-of select="$filename" />
    </xsl:if>
</xsl:template>
</xsl:stylesheet>
```

**XPath: rekursives
selektieren aller
<Ident>-
Elemente**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp. Kontrollstrukturen

Stringvergleiche mit `<xsl:if/>`:

Bvammer_PictureSelectDemo.xsl

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/xsl"
version="1.0">
<xsl:output method="xml"/>
<xsl:template match="/">
    <xsl:apply-templates select="/" />
</xsl:template>

<xsl:template match="Ident">
    <xsl:variable name="extension">.jpg</xsl:variable>
    <xsl:variable name="filename" select="./@PathName"/>
    <xsl:text>&#x0A;</xsl:text>
    <xsl:if test="substring($filename, string-length($filename)
- string-length($extension)+1) = $extension">
        <xsl:value-of select="./@PathName"/>
    </xsl:if>
</xsl:template>
</xsl:stylesheet>
```

**Abarbeitung des Ident-
Templates für alle
Ident-Elemente**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

<xsl:variable/>
<xsl:text/>
<xsl:if/>
substring / string-length
<xsl:value-of/>

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp. Kontrollstrukturen

Stringvergleiche mit <xsl:if/>:

Bvammer_PictureSelectDemo.xsl

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="xml"/>
  <xsl:template match="/">
    <xsl:apply-templates select="/" />
  </xsl:template>

  <xsl:template match="Ident">
    <xsl:variable name="extension">.jpg</xsl:variable>
    <xsl:variable name="filename" select="./@PathName"/>
    <xsl:text>&#x0A;</xsl:text>
    <xsl:if test="substring($filename, string-length($filename)
      - string-length($extension)+1) = $extension">
      <xsl:value-of select="./@PathName"/>
    </xsl:if>
  </xsl:template>
</xsl:stylesheet>
```

**Zwei Variablen für die
Dateiendung und den
eigentlichen Filenamen
im Attribut PathName**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

<xsl:variable/>
<xsl:text/>
<xsl:if/>
substring / string-length
<xsl:value-of/>

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp. Kontrollstrukturen

Stringvergleiche mit <xsl:if/>:

Bvammer_PictureSelectDemo.xsl

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="xml"/>
  <xsl:template match="/">
    <xsl:apply-templates select="*" />
  </xsl:template>

  <xsl:template match="Ident">
    <xsl:variable name="extension">.jpg</xsl:variable>
    <xsl:variable name="filename" select="./@PathName"/>
    <xsl:text>&#x0A;</xsl:text>
    <xsl:if test="substring($filename, string-length($filename)
      - string-length($extension)+1) = $extension">
      <xsl:value-of select="./@PathName"/>
    </xsl:if>
  </xsl:template>
</xsl:stylesheet>
```

Erzeugung eines
Zeilenumbruchs

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

<xsl:variable/>
<xsl:text/>
<xsl:if/>
substring / string-length
<xsl:value-of/>

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp. Kontrollstrukturen

Stringvergleiche mit <xsl:if/>:

Bvammer_PictureSelectDemo.xsl

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
  <xsl:output method="xml"/>
  <xsl:template match="/">
    <xsl:apply-templates select="*" />
  </xsl:template>

  <xsl:template match="Ident">
    <xsl:variable name="extension">.jpg</xsl:variable>
    <xsl:variable name="filename" select="./@PathName"/>
    <xsl:text>&#x0A;</xsl:text>
    <xsl:if test="substring($filename, string-length($filename)
      - string-length($extension)+1) = $extension">
      <xsl:value-of select="./@PathName"/>
    </xsl:if>
  </xsl:template>
</xsl:stylesheet>
```

**Trifft die Bedingung in der
test-Klausel zu, wird das
<value-of>-Element aus-
geführt**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

<xsl:variable/>

<xsl:text/>

<xsl:if/>

substring / string-length

<xsl:value-of/>

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp. Kontrollstrukturen

Stringvergleiche mit <xsl:if/>:

Bvammer_PictureSelectDemo.xsl

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
  <xsl:output method="xml"/>
  <xsl:template match="/">
    <xsl:apply-templates select="*" />
  </xsl:template>

  <xsl:template match="Ident">
    <xsl:variable name="extension">.jpg</xsl:variable>
    <xsl:variable name="filename" select="./@PathName"/>
    <xsl:text>&#x0A;</xsl:text>
    <xsl:if test="substring($filename, string-length($filename)
      - string-length($extension)+1) = $extension">
      <xsl:value-of select="./@PathName"/>
    </xsl:if>
  </xsl:template>
</xsl:stylesheet>
```

Stringvergleich: von der Variablen \$filename werden die letzten Zeichen der Länge des Strings in \$extension abgeschnitten mit der Xpath substring()-Funktion und dann verglichen.

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

<xsl:variable/>

<xsl:text/>

<xsl:if/>

substring / string-length

<xsl:value-of/>

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp. Kontrollstrukturen

Stringvergleiche mit <xsl:if/>:

Bvammer_PictureSelectDemo.xsl

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
  <xsl:output method="xml"/>
  <xsl:template match="/">
    <xsl:apply-templates select="*" />
  </xsl:template>

  <xsl:template match="Ident">
    <xsl:variable name="extension">.jpg</xsl:variable>
    <xsl:variable name="filename" select="./@PathName"/>
    <xsl:text>&#x0A;</xsl:text>
    <xsl:if test="substring($filename, string-length($filename)
      - string-length($extension)+1) = $extension">
      <xsl:value-of select="./@PathName"/>
    </xsl:if>
  </xsl:template>
</xsl:stylesheet>
```

**Leider gibt es keine
ends-with()-Funktion. Es gibt
aber eine
starts-with()-Funktion.**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

<xsl:variable/>
<xsl:text/>
<xsl:if/>
substring / string-length
<xsl:value-of/>

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Bsp. Kontrollstrukturen

Stringvergleiche mit <xsl:if/>:

Bvammer_PictureSelectDemo.xsl

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="xml"/>
  <xsl:template match="/">
    <xsl:apply-templates select="*" />
  </xsl:template>

  <xsl:template match="Ident">
    <xsl:variable name="extension">.jpg</xsl:variable>
    <xsl:variable name="filename" select="./@PathName"/>
    <xsl:text>&#x0A;</xsl:text>
    <xsl:if test="substring($filename, string-length($filename)
      - string-length($extension)+1) = $extension">
      <xsl:value-of select="./@PathName"/>
    </xsl:if>
  </xsl:template>
</xsl:stylesheet>
```

Der Dateiname wird nur dann in den Ergebnisbaum geschrieben, wenn der Test im <xsl:if/>-Element wahr (true) ist.

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

parameterübergabe shell
parameterübergabe php

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Sablotron Parameterübergabe (1)

- Man kann XSLT-Prozessoren Parameter übergeben, um das Verhalten des Prozessors von „außen“ zu steuern.
- Bei Sablotron geschieht dies wie folgt:

```
sabcmd <stylesheet> <input> param1=10 param2=30
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

parameterübergabe shell
parameterübergabe php

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Sablotron Parameterübergabe (2)

- Angenommen, wir wollen bewirken, dass nur bestimmte Dateinamen von dem Stylesheet verarbeitet werden
- Dann können wir dem Stylesheet in einer Variablen den übergebenen Dateinamen übergeben.
- Das Stylesheet entscheidet dann, ob es die Input-Datei bearbeitet oder nicht

- **Aufruf:**

```
sabcmd ParamDemo.xsl ParamDemo.xml  
'$fileName=ParamDemo.xml'
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

parameterübergabe shell
parameterübergabe php

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Sablotron Parameterübergabe (3)

- Der Aufruf:

```
sabcmd ParamDemo.xsl ParamDemo.xml  
'$fileName=ParamDemo.xml '
```

erzeugt folgende Ausgabe:

```
linuxsamsung:/SolutionDemos # sabcmd ParamDemo.xsl ParamDemo.xml '$fileName=test  
'  
Die Datei heisst test  
linuxsamsung:/SolutionDemos # _
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

parameterübergabe shell
parameterübergabe php

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Sablotron Parameterübergabe (4)

- Der Parameter `$fileName` wird im Stylesheet wie folgt angesprochen:

ParamDemo.xsl

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:output method="text" />

<xsl:param name="fileName" />

<xsl:template match="/">
    <xsl:text>Die Datei heisst</xsl:text>
    <xsl:value-of select="$fileName" />
    <xsl:text>&#x0A;</xsl:text>
</xsl:template>
</xsl:stylesheet>
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

parameterübergabe shell
parameterübergabe php

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Sablotron Parameterübergabe (4)

- Der Parameter `$fileName` wird im Stylesheet wie folgt angesprochen:

ParamDemo.xsl

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/T
<xsl:output method="text" />
<xsl:param name="fileName"/>
<xsl:template match="/">
    <xsl:text>Die Datei heisst</xsl:text>
    <xsl:value-of select="$fileName" />
    <xsl:text>&#x0A;</xsl:text>
</xsl:template>
</xsl:stylesheet>
```

**Parameter von aufrufender
Instanz (Shell oder PHP)
übernehmen**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

parameterübergabe shell
parameterübergabe php

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Sablotron Parameterübergabe (4)

- Der Parameter `$fileName` wird im Stylesheet wie folgt angesprochen:

ParamDemo.xsl

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/T
<xsl:output method="text" />

<xsl:param name="fileName" />

<xsl:template match="/">
    <xsl:text>Die Datei heisst</xsl:text>
    <xsl:value-of select="$fileName"/>
    <xsl:text>&#x0A;</xsl:text>
</xsl:template>
</xsl:stylesheet>
```

**Parameter selektieren und
in Ergebnisbaum/Dokument
schreiben**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

parameterübergabe shell
parameterübergabe php

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Sablotron Parameterübergabe aus dem PHP-Kontext

- Die Parameterübergabe aus dem PHP-Kontext funktioniert wie folgt:

ParamDemo.php

```
<?
$args = array('fileName' => 'ParamDemo.xml');

$xh = xslt_create();
$result = xslt_process($xh, "ParamDemo.xml", "ParamDemo.xsl",
NULL, array(), $args);

print($result);
xslt_free($xh);

?>
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

parameterübergabe shell
parameterübergabe php

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Sablotron Parameterübergabe aus dem PHP-Kontext

- Die Parameterübergabe aus dem PHP-Kontext funktioniert wie folgt:

ParamDemo.php

```
<?
$args = array('fileName' => 'ParamDemo.xml');

$xh = xslt_create();
$result = xslt_process($xh, "ParamDemo.xml", "ParamDemo.xsl",
NULL, array(), $args);

print($result);
xslt_free($xh);

?>
```

**In einem String-Array werden
die Argumentnamen und
Parameter übergeben**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

parameterübergabe shell
parameterübergabe php

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Sablotron Parameterübergabe aus dem PHP-Kontext

- Die Parameterübergabe aus dem PHP-Kontext funktioniert wie folgt:

ParamDemo.php

```
<?
$args = array('fileName' => 'ParamDemo.xml');

$xh = xslt_create();
$result = xslt_process($xh, "ParamDemo.xml", "ParamDemo.xsl",
NULL, array(), $args);

print($result);
xslt_free($xh);

?>
```

Und wie folgt übergeben

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

parameterübergabe shell
parameterübergabe php

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Sablotron Parameterübergabe aus dem PHP-Kontext

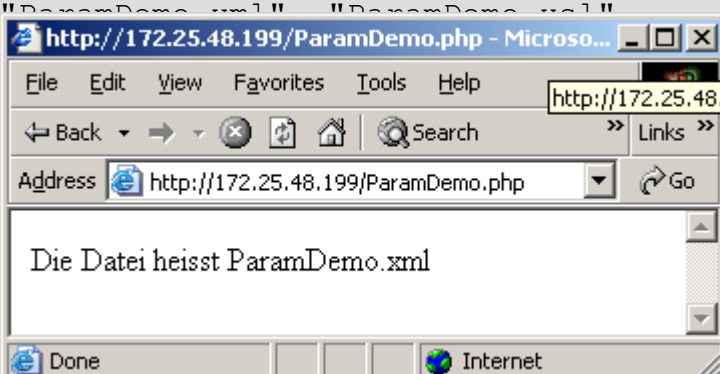
- Die Parameterübergabe aus dem PHP-Kontext funktioniert wie folgt:

```
ParamDemo.php
<?
$args = array('fileName' => 'ParamDemo.xml');

$xh = xslt_create();
$result = xslt_process($xh, "ParamDemo.xml", "ParamDemo.xml",
NULL, array(), $args);

print($result);
xslt_free($xh);

?>
```



The screenshot shows a browser window with the address bar containing 'http://172.25.48.199/ParamDemo.php'. The main content area displays the error message 'Die Datei heisst ParamDemo.xml'. The browser interface includes a menu bar (File, Edit, View, Favorites, Tools, Help), a search bar, and a status bar at the bottom showing 'Done' and 'Internet'.

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

**xpath-
funktionen I**

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Beispiel Datum (1)

- Datumsformate umwandeln:

DateDemo_1.xml

```
<?xml version="1.0" encoding="UTF-8"?>  
<?xml-stylesheet type="text/xsl"  
href="GermanDate2EnglishDate_1.xsl"?>  
  
<GermanDate>15.07.2003</GermanDate>
```



DateDemo_Result1.xml

```
<?xml version="1.0" encoding="UTF-16"?>  
<EnglishDate>2003-07-15</EnglishDate>
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

Beispiel Datum (2)

- Wie sieht das Stylesheet dazu aus?

GermanDate2EnglishDate_1.xsl

```
<?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"/>

<xsl:template match="/">
  <xsl:element name="EnglishDate">
    <xsl:variable name="TheDateVar" select="./GermanDate"/>

    <xsl:value-of select=
      "concat(substring($TheDateVar,7,4),'-',
        substring($TheDateVar,4,2),'-',
        substring($TheDateVar,1,2) ) "/>

  </xsl:element>
</xsl:template>
</xsl:stylesheet>
```

Position im String

Anzahl Buchstaben

1	5	.	0	7	.	2	0	0	3
---	---	---	---	---	---	---	---	---	---

1 2 3 4 5 6 7 8

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

<xsl:call-tenmplate/>
<xsl:include/>

xpath-
funktionen II

xslt-funktionen

Beispiel DateDemoTransform

- Für den Fall, das wir ein Template mehrfach gebrauchen können und es explizit aufrufen wollen, können wir das XSLT-Element

`<xsl:call-template/>` nutzen.

- Das Template muss dann mit dem Namensattribut **name** definiert werden.

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

<xsl:call-template/>
<xsl:include/>

xpath-
funktionen II

xslt-funktionen

Beispiel DateDemoTransform

- Wie sieht das Stylesheet dazu aus?

DateDemoTransform.xsl

```
...
<xsl:template match="/">
<xsl:element name="EnglishDate">
<xsl:variable name="TheDateVar" select="./GermanDate"/>
    <xsl:call-template name="g2eDate">
        <xsl:with-param name="dateStr"
            select="$TheDateVar"/>
    </xsl:call-template>
</xsl:element>
</xsl:template>

<xsl:template name="g2eDate">
    <xsl:param name="dateStr"/>
    <xsl:value-of select=
        "concat(substring($dateStr,7,4),'-',
            substring($dateStr,4,2),'-',
            substring($dateStr,1,2) ) "/>
</xsl:template>
...
```

Der Datum-String wird in
die Variable TheDateVar
geschrieben

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

<xsl:call-template/>
<xsl:include/>

xpath-
funktionen II

xslt-funktionen

Beispiel DateDemoTransform

- Wie sieht das Stylesheet dazu aus?

DateDemoTransform.xsl

```
...  
<xsl:template match="/">  
<xsl:element name="EnglishDate">  
<xsl:variable name="TheDateVar" select="./GermanDate"/>  
  <xsl:call-template name="g2eDate">  
    <xsl:with-param name="dateStr"  
      select="$TheDateVar"/>  
  </xsl:call-template>  
</xsl:element>  
</xsl:template>
```

```
<xsl:template name="g2eDate">  
  <xsl:param name="dateStr"/>  
  <xsl:value-of select=  
    "concat(substring($dateStr,7,4),'-',  
      substring($dateStr,4,2),'-',  
      substring($dateStr,1,2) ) "/>  
</xsl:template>
```

Anschließend wird mittels
<xsl:call-template> das
Untere Template g2eDate
aufgerufen

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

<xsl:call-template/>
<xsl:include/>

xpath-
funktionen II

xslt-funktionen

Beispiel DateDemoTransform

- Wie sieht das Stylesheet dazu aus?

DateDemoTransform.xsl

```
...  
<xsl:template match="/">  
<xsl:element name="EnglishDate">  
<xsl:variable name="TheDateVar" select="./GermanDate"/>  
  <xsl:call-template name="g2eDate">  
    <xsl:with-param name="dateStr"  
      select="$TheDateVar"/>  
  </xsl:call-template>  
</xsl:element>  
</xsl:template>  
  
<xsl:template name="g2eDate">  
  <xsl:param name="dateStr"/>  
  <xsl:value-of select=  
    "concat(substring($dateStr,7,4),'-',  
      substring($dateStr,4,2),'-',  
      substring($dateStr,1,2) ) "/>  
</xsl:template>  
...
```

Dabei wird dem aufgerufenen Template das Datum mitgegeben

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

<xsl:call-template/>
<xsl:include/>

xpath-
funktionen II

xslt-funktionen

Beispiel DateDemoTransform

- Wie sieht das Stylesheet dazu aus?

DateDemoTransform.xsl wird von DateDemo_3.xml aufgerufen

```
...  
<xsl:template match="/">  
<xsl:element name="EnglishDate">  
<xsl:variable name="TheDateVar" select="./GermanDate"/>  
  <xsl:call-template name="g2eDate">  
    <xsl:with-param name="dateStr"  
      select="$TheDateVar"/>  
  </xsl:call-template>  
</xsl:element>  
</xsl:template>  
  
<xsl:template name="g2eDate">  
  <xsl:param name="dateStr"/>  
  <xsl:value-of select=  
    "concat(substring($dateStr,7,4),'-',  
      substring($dateStr,4,2),'-',  
      substring($dateStr,1,2) ) "/>  
</xsl:template>  
...
```

**Der Rest läuft wie im
vorigen Beispiel ab.**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

<xsl:call-template/>
<xsl:include/>

xpath-
funktionen II

xslt-funktionen

<xsl:include href="" />

- Solche eigenen Erweiterungen kann man auslagern

DateDemoTransform2.xsl

```
...  
<xsl:include  
href="MyExtensions.xsl"/>  
<xsl:call-template name="g2eDate">  
<xsl:with-param name="dateStr" select="$TheDateVar"/>  
</xsl:call-template>
```

MyExtensions.xsl wird
eingebunden

MyExtensions.xsl

```
...  
<xsl:template name="g2eDate">  
  <xsl:param name="dateStr"/>  
  <xsl:value-of select=  
    "concat(substring($dateStr,7,4),'-',  
      substring($dateStr,4,2),'-',  
      substring($dateStr,1,2) ) "/>  
</xsl:template>
```

...

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

<xsl:call-template/>
<xsl:include/>

xpath-
funktionen II

xslt-funktionen

<xsl:include href="" />

- Solche eigenen Erweiterungen kann man auslagern

DateDemoTransform2.xsl

```
...  
<xsl:include  
href="MyExtensions.xsl"/>  
<xsl:call-template name="g2eDate">  
<xsl:with-param name="dateStr" select="$TheDateVar"/>  
</xsl:call-template>
```

Und dann aufgerufen

MyExtensions.xsl

```
...  
<xsl:template name="g2eDate">  
  <xsl:param name="dateStr"/>  
  <xsl:value-of select=  
    "concat(substring($dateStr,7,4),'-',  
    substring($dateStr,4,2),'-',  
    substring($dateStr,1,2))"/>  
</xsl:template>
```

...

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

count
sum
div
round

xslt-funktionen

Beispiel Rechnen mit XPath

- Wir wollen aus folgender XML-Struktur den Durchschnittspreis über alle Bücher berechnen.

BuchDemo_1.xml

```
<buecher>
  <buch titel="Den Wald vor lauter Bäumen sehen">
    <autor>Dennis Sheerwood</autor>
    <preis>40.00</preis>
  </buch>
  <buch titel="Die Kunst vernetzt zu denken">
    <autor>Frederic Vester</autor>
    <preis>38.20</preis>
  </buch>
  .
  .
</buecher>
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

count
sum
div
round

xslt-funktionen

Beispiel Rechnen mit XPath

- Das passende Stylesheet dazu könnte so aussehen:

BuchPreis_1.xsl

```
<xsl:template match="/">  
  <xsl:variable name="NumOfBooks" select="count(//preis)"/>  
  <xsl:variable name="theSum" select="sum(//preis)"/>  
  <xsl:value-of select="round($theSum div $numOfBooks)"/>  
</xsl:template>
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

count
sum
div
round

xslt-funktionen

Beispiel Rechnen mit XPath

- Das passende Stylesheet dazu könnte so aussehen:

BuchPreis_1.xsl

```
<xsl:template match="/">  
  <xsl:variable name="NumOfBooks" select="count(//preis)"/>  
  <xsl:variable name="theSum" select="sum(//preis)"/>  
  <xsl:value-of select="round($theSum div $numOfBooks)"/>  
</xsl:template>
```

Alle <preis>-Elemente unterhalb der Wurzel Selektieren und zählen mit der XPath-Funktion count(). Beachte Rekursivoperator // (!).

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

count
sum
div
round

xslt-funktionen

Beispiel Rechnen mit XPath

- Das passende Stylesheet dazu könnte so aussehen:

BuchPreis_1.xsl

```
<xsl:template match="/">  
  <xsl:variable name="NumOfBooks" select="count(//preis)"/>  
  <xsl:variable name="theSum" select="sum(//preis)"/>  
  <xsl:value-of select="round($theSum div $numOfBooks)"/>  
</xsl:template>
```

**Alle <preis>-Elemente
unterhalb der Wurzel
Selektieren und aufsummieren
mit der XPath-Funktion `sum()`.
Beachte Rekursivoperator `//` (!).**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

count
sum
div
round

xslt-funktionen

Beispiel Rechnen mit XPath

- Das passende Stylesheet dazu könnte so aussehen:

BuchPreis_1.xsl

```
<xsl:template match="/">  
  <xsl:variable name="NumOfBooks" select="count(//preis)"/>  
  <xsl:variable name="theSum" select="sum(//preis)"/>  
  <xsl:value-of select="round($theSum div $numOfBooks)"/>  
</xsl:template>
```

Die Variable **\$theSum** dividieren
durch **\$numOfBooks**.
**Beachte Operator div für
Division (!)**.

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

count
sum
div
round

xslt-funktionen

Beispiel Rechnen mit XPath

- Das passende Stylesheet dazu könnte so aussehen:

BuchPreis_1.xsl

```
<xsl:template match="/">  
  <xsl:variable name="NumOfBooks" select="count(//preis)"/>  
  <xsl:variable name="theSum" select="sum(//preis)"/>  
  <xsl:value-of select="round($theSum div $numOfBooks)"/>  
</xsl:template>
```

Und diesen Wert runden mit der
round()-Funktion.

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

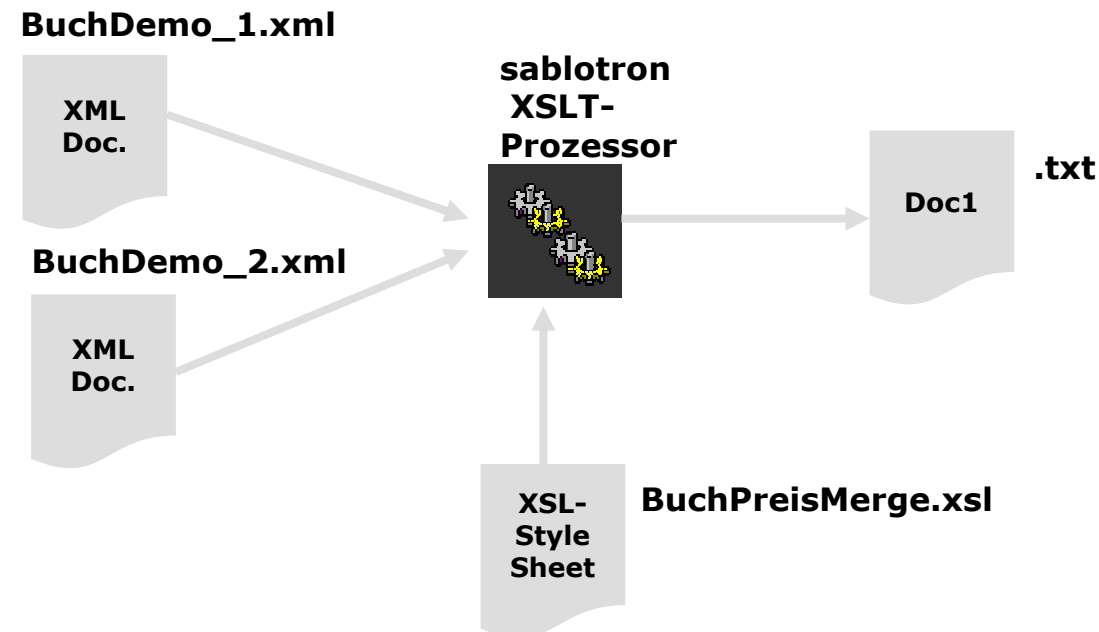
xpath-
funktionen II

xslt-funktionen

document
extensions

Dateien Verknüpfen mit der document()-Funktion

- Können wir aus verschiedenen XML-Quellen mehrere Dateien zusammenführen und mit dem Prozessor verarbeiten?



überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

Dateien Verknüpfen mit der document()-Funktion

- Schauen wir uns zunächst unsere beiden verschiedenen XML-Strukturen an:

BuchDemo_1.xml

```
<buecher>
  <buch titel="Den Wald vor lauter Bäumen sehen">
    <autor>Dennis Sheerwood</autor>
    <preis>40.00</preis>
  </buch>
  <buch titel="Die Kunst vernetzt zu denken">
    <autor>Frederic Vester</autor>
    <preis>38.20</preis>
  </buch>
  <buch titel="Der Termin">
    <autor>Tom DeMarco</autor>
    <preis>19.88</preis>
  </buch>
</buecher>
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

Dateien Verknüpfen mit der document()-Funktion

- Schauen wir uns zunächst unsere beiden verschiedenen XML-Strukturen an:

BuchDemo_2.xml

```
<books>
  <book title="XSLT">
    <author>Doug Tidwell</author>
    <publisher>O'REILLY</publisher>
    <prize>40</prize>
  </book>
  <book title="XSLT">
    <author>Michael Kay</author>
    <publisher>Wrox</publisher>
    <prize>34.99</prize>
  </book>
  .
  .
</books>
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

Dateien Verknüpfen mit der document()-Funktion

- BuchDemo_1.xml soll normal verarbeitet werden, indem wir die Datei dem XSLT-Prozessor über den Aufruf einfach mitgeben
- Der Name der Datei BuchDemo_2.xml wird über eine zusätzlichen Parameter übergeben, sodass im XSL-Stylesheet BuchPreisMerge.xsl über die document()-Funktion darauf zugegriffen werden kann.

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

Dateien Verknüpfen mit der document()-Funktion

- Hier das passende Stylesheet dazu:

BuchPreisMerge.xsl

```
<xsl:param name="fileName"/>
<xsl:template match="/">
  <xsl:variable name="numOfBooks" select="count(//buch)"/>
  <xsl:variable name="theSum" select="sum(//preis)"/>
  ...
  <xsl:value-of select="round($theSum div $numOfBooks)"/>

  <xsl:variable name="numOfExternalBooks"
select="count(document($fileName)/books//book)"/>

  <xsl:variable name="sumOfExternalBooks"
select="sum(document($fileName)/books//prize)"/>
  <xsl:value-of select="round($sumOfExternalBooks div
$numOfExternalBooks)"/>
</xsl:template>
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

Dateien Verknüpfen mit der document()-Funktion

- Hier das passende Stylesheet dazu:

BuchPreisMerge.xsl

```
<xsl:param name="fileName"/>
<xsl:template match="/">
  <xsl:variable name="numOfBooks" select="count(//buch)"/>
  <xsl:variable name="theSum" select="sum(//preis)"/>
  ...
  <xsl:value-of select="round($theSum div $numOfBooks)"/>

  <xsl:variable name="numOfExtBooks"
select="count(document($file
...
  <xsl:variable name="sumOfExtBooks"
select="sum(document($file
...
  <xsl:value-of select="round
($sumOfExtBooks div
$numOfExternalBooks)"/>
</xsl:template>
```

**Zunächst holen wir uns mit dem
Rekursiv-Operator // die Anzahl
der Bücher mit den <buch>-
Elementen, zählen diese mit
der count()-Funktion und
schreiben das Ergebnis in die
Variable \$numOfBooks**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

Dateien Verknüpfen mit der document()-Funktion

- Hier das passende Stylesheet dazu:

BuchPreisMerge.xsl

```
<xsl:param name="fileName"/>
<xsl:template match="/">
  <xsl:variable name="numOfBooks" select="count(//buch)"/>
  <xsl:variable name="theSum" select="sum(//preis)"/>
  ...
  <xsl:value-of select="round($theSum div $numOfBooks)"/>

  <xsl:variable name="numOfExtBooks"
  select="count(document($fileName)/*)"/>

  <xsl:variable name="sumOfExtBooks"
  select="sum(document($fileName)/*)"/>
  <xsl:value-of select="round(
  $sumOfExtBooks div $numOfExternalBooks)"/>
</xsl:template>
```

**Dann ermitteln wir die Summe
Aller Buchpreise mit dem
Rekursivoperator // von der
Wurzel (!) Die Summe wird mit
der sum()-Funktion gebildet und
in die Variable \$sumOfBooks
geschrieben.**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

Dateien Verknüpfen mit der document()-Funktion

- Hier das passende Stylesheet dazu:

BuchPreisMerge.xsl

```
<xsl:param name="fileName"/>
<xsl:template match="/">
  <xsl:variable name="numOfBooks" select="count(//buch)"/>
  <xsl:variable name="theSum" select="sum(//preis)"/>
  ...
  <xsl:value-of select="round($theSum div $numOfBooks)"/>

  <xsl:variable name="numOfExternalBooks" select="count(document($file...))"/>
  <xsl:variable name="sumOfExternalBooks" select="sum(document($file...))"/>
  <xsl:value-of select="round($sumOfExternalBooks div $numOfExternalBooks)"/>
</xsl:template>
```

**Nun wird wie im vorherigen
Beispiel noch die Summe durch
die Anzahl der Bücher dividiert,
gerundet und das Resultat in den
Ergebnisbaum geschrieben**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

Dateien Verknüpfen mit der document()-Funktion

- Hier das passende Stylesheet dazu:

BuchPreisMerge.xsl

```
<xsl:param name="fileName" />
<xsl:template match="/">
  <xsl:variable name="numOfBooks" select="count(//book)" />
  <xsl:variable name="theS
  ...
  <xsl:value-of select="ro

  <xsl:variable name="numO
  select="count (document ($

  <xsl:variable name="sumOfExternalBooks"
  select="sum (document ($fileName) /books//prize) " />
  <xsl:value-of select="round ($sumOfExternalBooks div
  $numOfExternalBooks) " />
</xsl:template>
```

Schauen wir uns jetzt die XML-Struktur in der zweiten XML-Datei an. Der Name der 2. Datei BuchDemo_2.xml steht in der Variable \$fileName, die beim Prozessor-Aufruf mit übergeben wird.

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

Dateien Verknüpfen mit der document()-Funktion

- Hier das passende Stylesheet dazu:

BuchPreisMerge.xsl

```
<xsl:param name="fileName"/>
<xsl:template match="/">
  <xsl:variable name="numOfBooks" />
  <xsl:variable name="theSum" />
  ...
  <xsl:value-of select="round($sumOfExternalBooks div $numOfExternalBooks)"/>

  <xsl:variable name="numOfExternalBooks"
    select="count(document($fileName)/books//book)"/>

  <xsl:variable name="sumOfExternalBooks"
    select="sum(document($fileName)/books//prize)"/>
  <xsl:value-of select="round($sumOfExternalBooks div $numOfExternalBooks)"/>
</xsl:template>
```

**Über die document()-Funktion
Können wir direkt auf den Baum
Der Datei wie gewohnt zugreifen.
Die Funktion bekommt einen
Filenames oder eine Url mit
übergeben.**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xp
fu
we
fu
xp
fu

Diesmal ist der Zugriff geschickter, da wir rekursiv auf alle Bücher unter /books zugreifen und nicht rekursiv von der Wurzel. Die andere Variante könnte zu Fehler führen, falls <buch>-Elemente im Baum unter einer Struktur enthalten sind, die aus irgendwelchen Gründen nicht mitgezählt werden sollen.

XSLT-Funktionen

document
extensions

Dateien Verknüpfen mit der document()-Funktion

- Hier das passende Stylesheet dazu:

BuchPreisMerge.xsl

```
me="fileName"/>
  match="/">
    xsl:variable name="numOfBooks" select="count(//buch)"/>
    xsl:variable name="theSum" select="sum(//preis)"/>
    xsl:value-of select="round($theSum div $numOfBooks)"/>
    xsl:variable name="numOfExternalBooks"
      count(document($fileName)/books//book)"/>
    xsl:variable name="sumOfExternalBooks"
      select="sum(document($fileName)/books//prize)"/>
    <xsl:value-of select="round($sumOfExternalBooks div
      $numOfExternalBooks)"/>
  </xsl:template>
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

Dateien Verknüpfen mit der document()-Funktion

- Hier das passende Stylesheet dazu:

BuchPreisMerge.xsl

```
<xsl:param name="fileName"/>
<xsl:template match="/">
  <xsl:variable name="numOfBooks" select="count(document($fileName)/books//book)"/>
  <xsl:variable name="theSum" select="sum(document($fileName)/books//prize)"/>
  ...
  <xsl:value-of select="round($theSum div $numOfBooks)"/>

  <xsl:variable name="numOfExternalBooks"
    select="count(document($fileName)/books//book)"/>

  <xsl:variable name="sumOfExternalBooks"
    select="sum(document($fileName)/books//prize)"/>
  <xsl:value-of select="round($sumOfExternalBooks div
    $numOfExternalBooks)"/>
</xsl:template>
```

**Analog besorgen wir uns auch
die Summe über alle engl.
Buchpreistags <prize> und
dividieren wie zuvor.**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

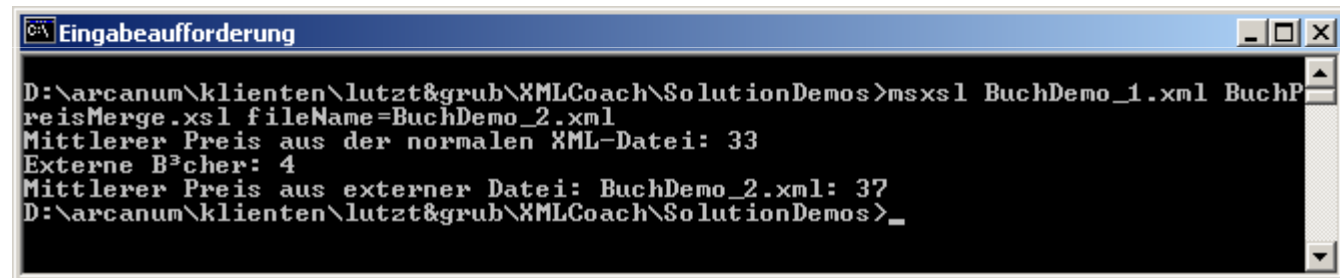
xpath-
funktionen II

xslt-funktionen

document
extensions

Dateien Verknüpfen mit der document()-Funktion

- Der Aufruf mit dem Microsoft XSLT-Prozessor sieht wie folgt aus:



```
D:\arcanum\klienten\lutz&grub\XMLCoach\SolutionDemos>msxsl BuchDemo_1.xml BuchP  
reisMerge.xsl fileName=BuchDemo_2.xml  
Mittlerer Preis aus der normalen XML-Datei: 33  
Externe Bücher: 4  
Mittlerer Preis aus externer Datei: BuchDemo_2.xml: 37  
D:\arcanum\klienten\lutz&grub\XMLCoach\SolutionDemos>_
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

Mehr Funktionalität?

- Ab XSLT V1.1 können externe Funktionen in einem XSLT-Stylesheet aufgerufen werden
- Dies können JavaScript oder Java oder andere Erweiterungen sein
- Im Sprachgebrauch sind die die XSLT-Extensions
- Sablotron verfügt über eine JavaScript-Anbindung

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

Mehr Funktionalität?

- Die Extensions haben nur lesenden Zugriff auf das Document Object Model (DOM)
- Unter www.exslt.org findet man einige JavaScript-Module als XSLT-Module
- Man kann JavaScript-Funktionen jedoch auch direkt einbetten
- Sablotron nutzt dafür den JavaScript-Interpreter vom Mozilla (www.mozilla.org/js)

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

XSLT-Extensions (1)

- Hier das passende Stylesheet dazu:

JavaScriptDemo.xsl

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/
  1999/XSL/Transform"
  xmlns:func="http://www.exslt.org/functions"
  xmlns:my="http://gingerall.org/sablot/myfunc"
  extension-element-prefixes="func">
<xsl:output method="text"/>

<func:script implements-prefix="my" language="javascript">
<![CDATA[
  function attrSum(nodeset) {
    var sum = new Date().toString();
    return sum;
  } ]]>
</func:script>
....
```

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

XSLT-Extensions (1)

- Hier das passende Stylesheet dazu:

JavaScriptDemo.xsl

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:func="http://www.exslt.org/functions"
  xmlns:my="http://gingerall.org/sablot/myfunc"
  extension-element-prefixes="func">
<xsl:output method="text"/>
```

```
<func:script implements-prefix="func">
<![CDATA[
  function attrSum(nodeset)
    var sum = new Date().toLocaleString();
    return sum;
  ] ]>
</func:script>
....
```

**Zunächst müssen wir einen
XML-Namensraum (namespace)
für die Extensions definieren (!)**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

XSLT-Extensions (1)

- Hier das passende Stylesheet dazu:

JavaScriptDemo.xsl

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:func="http://www.exslt.org/functions"
  xmlns:my="http://gingerall.org/sablot/myfunc"
  extension-element-prefixes="func">
<xsl:output method="text"/>
```

```
<func:script implements-prefix="my">
<![CDATA[
  function attrSum(nodeset)
    var sum = new Date().toLocaleDateString();
    return sum;
  ] ]>
</func:script>
....
```

**Dann benötigen wir noch einen
Namensraum für die Funktionen
selbst (!)**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

XSLT-Extensions (1)

- Hier das passende Stylesheet dazu:

JavaScriptDemo.xsl

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:func="http://www.exslt.org/functions"
  xmlns:my="http://gingerall.org/sablot/myfunc"
  extension-element-prefixes="func">
<xsl:output method="text"/>
```

```
<func:script implements-prefix="func">
<![CDATA[
  function attrSum(nodeset)
    var sum = new Date().toLocaleDateString();
    return sum;
  } ]]>
</func:script>
....
```

**Über das Attribut
extension-element-prefixes
weisen wir letztlich zu, unter
welchem Namensraum-Prefix
der Prozessor die Extensions
findet**

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

XSLT-Extensions (1)

- Hier das passende Stylesheet dazu:

```
JavaScriptDemo.xsl
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:func="http://www.w3.org/2005/xsl/functions"
  xmlns:my="http://ginger.de/xslt/extension-element-prefix"
  extension-element-prefix="my"
  <xsl:output method="text"/>

<func:script implements-prefix="my" language="javascript">
<![CDATA[
  function attrSum(nodeset) {
    var sum = new Date().toString();
    return sum;
  } ]]>
</func:script>
....
```

Dann können wir den neuen Namensraum nutzen, um unsere JavaScript-Funktionen zu implementieren.

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

XSLT-Extensions (1)

- Hier das passende Stylesheet dazu:

```
JavaScriptDemo.xsl
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:func="http://www.w3.org/2005/xsl/functions"
  xmlns:my="http://ginger.de/xslt/extension-element-prefix" />
<xsl:output method="text"/>

<func:script implements-prefix="my" language="javascript">
<![CDATA[
  function attrSum(nodeset) {
    var sum = new Date().toString();
    return sum;
  } ]]>
</func:script>
....
```

Die Funktion wird in einem
CDATA (Character-Data)-Feld
„gewrappt“, damit dieser nicht
geparst wird (!)

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

XSLT-Extensions (1)

- Hier das passende Stylesheet dazu:

```
JavaScriptDemo.xsl
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:func="http://www.w3.org/2005/xsl/functions"
  xmlns:my="http://ginger.com/xslt/extension-element-prefix"
  extension-element-prefix="my"
  <xsl:output method="text"/>

<func:script implements-prefix="my" language="javascript">
<![CDATA[
  function attrSum() {
    var d = new Date().toString();
    return d;
  } ]]>
</func:script>
....
```

Es folgt ganz normaler ECMA bzw. JavaScript-Code. Diese Funktion liest über das Betriebssystem das aktuelle Datum und Uhrzeit auf, indem ein Date-Object instantiiert wird.

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weitere xslt-
funktionen

xpath-
funktionen II

xslt-funktionen

document
extensions

XSLT-Extensions (1)

- Hier das passende Stylesheet dazu:

JavaScriptDemo.xsl

```
...  
<xsl:template match="/">  
    <xsl:text>This file was processed on </xsl:text>  
    <xsl:value-of select="my:attrSum()" />  
    <xsl:text> by an XSLT-Processor&#x0a; </xsl:text>  
</xsl:template>  
...
```

Aufgerufen wird die JavaScript-Funktion wie eine XPath-Funktion, mit dem zuvor festgelegten Namensraum für die eigenen Funktionen (!)

überblick und
architektur

xslt und xpath

xpath axes

xslt kontroll-
strukturen

sablotron

xpath-
funktionen I

weite
funkt

xpath
funktionen II

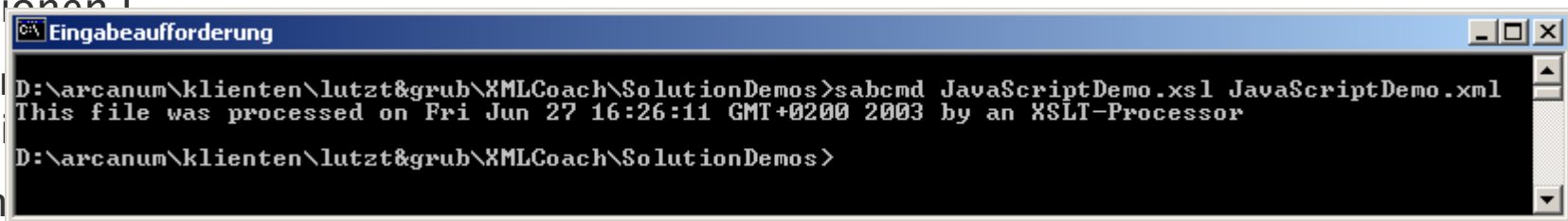
xslt-funktionen

document

extensions

XSLT-Extensions (2)

- Der Aufruf und die Ausgabe zu dem Stylesheet sieht wie folgt aus:



```
Eingabeaufforderung
D:\arcanum\klienten\lutz&grub\XMLCoach\SolutionDemos>sabcmd JavaScriptDemo.xsl JavaScriptDemo.xml
This file was processed on Fri Jun 27 16:26:11 GMT+0200 2003 by an XSLT-Processor
D:\arcanum\klienten\lutz&grub\XMLCoach\SolutionDemos>
```