Medien-
Technik

XML eXtensibel Markup Language

Entwicklung seit 1996 im W3C
http://www.w3.org/pub/WWW/TR/WD-xml-961114.html ;

„Extensible Markup Language (XML) is an extremely simple dialect of SGML... The goal is to enable generic SGML to be served, received, and processed on the Web in the way that is now possible with HTML. For this reason, XML has been designed for ease of implementation, and for interoperability with both SGML and HTML.“

(1st Working Draft)

Medien-
Technik

Entwurfsziele von XML

The design goals for XML are:

1. XML shall be straightforwardly usable over the Internet.
2. XML shall support a wide variety of applications.
3. XML shall be compatible with SGML.
4. It shall be easy to write programs which process XML documents.
5. The number of optional features in XML is to be kept to the absolute minimum, ideally zero.
6. XML documents should be human-legible and reasonably clear.
7. The XML design should be prepared quickly.
8. The design of XML shall be formal and concise.
9. XML documents shall be easy to create.
10. Terseness is of minimal importance.

terseness
Knappheit
Prägnanz

Medien-
Technik

XML Grungedanken

- XML beschreibt **Dokumente, genauer ihre logische Struktur** bzw. dient dem Austausch von Daten
- XML ist keine Erweiterung von HTML
- XML ist eine Untermenge von SGML in dem Sinne, dass jedes XML-Dokument ein SGML-Dokument ist.
- XML tut nichts
- XML hat Tags wie HTML
  <tag> … </tag>
- XML-Tags immer öffnend und schließend (Klammerstruktur)
- XML-Tags sind frei definierbar
- XML-Tags unterscheiden Groß-Kleinschreibung
- <Tag> und <tag> sind verschieden.

Medien-
Technik

XML Beispiel

Jedes XML-Dokument muss genau ein Wurzel-Element haben!
Das Dokument hat Baumstruktur.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note>
 <to>Tove</to>
 <from>Jani</from>
 <heading>Reminder</heading>
 <body>Don't forget me this weekend!</body>
</note>
```

Alle XML-Elemente

- müssen ein Abschluss-Tag haben
- müssen eine intakte Klammerstruktur bilden
- unterscheiden Groß-und Kleinschreibung

Medien-
Technik

XML Beispiel

Tag-Attribute immer in Anführung!

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note date="12/11/2002">
 <to>Tove</to>
 <from>Jani</from>
<body>Don't
forget        me
this weekend!</body>
</note>
```

Anders als in HTML haben Leerzeichen und Zeilenumbrüche Bedeutung!
Aus CR LF wird als LF.

Medien-
Technik

XML Beispiel

XML-Dokumente sind erweiterbar. (Ignorieren oder …)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note>
<date>12/11/2002</date>
 <to>Tove</to>
 <from>Jani</from>
<body>Don't forget me this weekend!</body>
</note>
```

Elementnamen können aus Buchstaben, Ziffern und sonstigen Zeichen bestehen
Sie dürfen nicht mit „xml“, „Xml“, „XML“ beginnen.
Sie dürfen nicht mit einer Ziffer oder einem Sonderzeichen beginnen.

Medien-
Technik

SVG
Scalable
Vector
Graohics

```
<svg width="4in" height="3in">
<g>
<rect x="50" y="80" width="200"
   height="300" style= "fill:#FFFFCC" />
</g>
```

MathML
Mathematical
Markup
Language

```
<msup>
 <mfenced>
  <mrow>
  <mi>a</mi> <mo>*</mo>
<mi>b</mi>
  </mrow>
 </mfenced>
 <mn>2<mn>
</msup>
```

$$(a+b)^2$$

Medien-
Technik

SMIL
Synchronised
Multimedia
Integration
Language

```
<par>
 <text src="title.rt" type="text/html"
    region="title" duration="20s"/>
 <audio src=„"Intro.wav" />
</par>
```

MathML
Mathematical
Markup
Language

```
<msup>
 <mfenced>
  <mrow>
  <mi>a</mi> <mo>*</mo>
<mi>b</mi>
  </mrow>
 </mfenced>
 <mn>2<mn>
</msup>
```

$$(a+b)^2$$

Medien-
Technik

XML Document Type Definition

XML-Dokumente mit korrekter Syntax heißen „well formed“
XML-Dokumente mit zugehöriger DTD heißen „valid“
Validierung anhand einer DTD

Document Type kann im Dokument selbst enthalten sein

```
<?xml version="1.0"?>
<!DOCTYPE note [
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>
```

```
<note>
 <to>Tove</to>
 <from>Jani</from>
```


cd_catalog_with_css.xml

PCDATA parseable character data

to parse
einen Satz grammatisch zergliedern

Medien-
Technik

XML DTD


```
<?xml version="1.0"?>
<!DOCTYPE note SYSTEM "note.dtd">
<note>
 <to>Tove</to>
 <from>Jani</from>
 <heading>Reminder</heading>
 <body>Don't forget me this weekend!</body>
</note>
```

Document Type kann in getrennter Datei (URL) enthalten sein

note.dtd

```
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
```

Medien-
Technik

XML-Dokumente bestehen aus folgenden einfachen Bausteinen

- **Elemente**

  Ein Dokument besteht aus einem Element, das weitere Elemente enthalten kann.
- **Tags**

  Markups, Auszeichnungen, Marken, die Elemente begrenzen
- **Attribute**

  Ergänzende Information für Elemente
- **PCDATA**

  Parsed character data, wird von XML-Parsern interpretiert
- **CDATA**

  Zeichenfolge, die nicht weiter analysiert werden muss
- <!-- This is a comment -->

cd-catalog.xml

cd-catalog_with_css.xml

Medien-
Technik

XML DTD


cd_catalog_with_css.xml

- **Element-Deklaration**

(1) <!ELEMENT element-name category>
oder
(2) <!ELEMENT element-name (element-content)>

Beispiele (1):

<!ELEMENT br EMPTY>

Verwendung:

kurz für

</br>

<!ELEMENT note ANY>

Verwendung:
<note> …. </note>

Medien-
Technik

XML DTD


cd_catalog_with_css.xml

- **Element-Deklaration**

(2) <!ELEMENT element-name (element-content)>

<!ELEMENT to (#PCDATA)>

<!ELEMENT note (to,from,heading,body)>

In dieser Reihenfolge

<!ELEMENT to (message)> genau eine message

<!ELEMENT to (message+)> mindestens 1 message

<!ELEMENT to (message*)> mindestens 0 message

<!ELEMENT to (message?)> 0 oder 1 message

<!ELEMENT note (to,from,header,(message|body))>
alternativ message oder body

<!ELEMENT note
(#PCDATA|to|from|header|message)*>

Medien-
Technik

XML DTD


cd_catalog_with_css.xml

- Attributlisten
- <!ATTLIST element-name attribute-name attribute-type default-value>
- attribute-type
  - CDATA
  - (string1|string2|..) Aufzählungtyp
  - ID eindeutiger Bezeichner
  - ...

Beispiele:

<!ATTLIST person sex (male|femal) “male“>

<person sex=‘male‘>

Medien-
Technik

XML DTD


cd_catalog_with_css.xml

- Entities
  - Sind Kürzel, benannte Zeichen (folgen)
  - Verallgemeinerung ä etc. in HTML
  - Syntax <!ENTITY entity-name “entity-value“>
  - Beispiel:
  - <!ENTITY Vorlesung “Medientechnik 2007-8“>
  - Verwendung
  - Im Rahmen der &Vorlesung;

Medien-
Technik

XML-DTD: Beispiel TV-Programm

```
<!DOCTYPE TVSCHEDULE [  <!ELEMENT TVSCHEDULE (CHANNEL+)>
    <!ELEMENT CHANNEL (BANNER, DAY+)>
    <!ELEMENT BANNER (#PCDATA)>
    <!ELEMENT DAY ((DATE, HOLIDAY) | (DATE, PROGRAMSLOT+))+>
    <!ELEMENT HOLIDAY (#PCDATA)>
    <!ELEMENT DATE (#PCDATA)>
    <!ELEMENT PROGRAMSLOT (TIME, TITLE, DESCRIPTION?)>
    <!ELEMENT TIME (#PCDATA)>
    <!ELEMENT TITLE (#PCDATA)>
    <!ELEMENT DESCRIPTION (#PCDATA)>
<!ATTLIST TVSCHEDULE NAME CDATA #REQUIRED>
    <!ATTLIST CHANNEL CHAN CDATA #REQUIRED>
    <!ATTLIST PROGRAMSLOT VTR CDATA #IMPLIED>
    <!ATTLIST TITLE RATING CDATA #IMPLIED>
    <!ATTLIST TITLE LANGUAGE CDATA #IMPLIED>
]>
```

Medien-
Technik

XML-DTD: Beispiel TV-Programm

Channel

Sender: zur Programmübersicht

ZDF

Programm auf **ZDF** Am Mittwoch, dem 21.11.2007

Uhrzeit:	Sendung:	Kategorie:
05:30	ZDF-Morgenmagazin	Magazin/Infomagazin
09:00	heute	Magazin/Nachrichten
09:05	**Volle Kanne - Service täglich**	
10:00	heute	Magazin/Nachrichten
10:30	Wege zum Glück	Serie/Telenovela
11:15	Reich und Schön	Serie/Familien
11:35	Reich und Schön	Serie/Familien

www.tvinfo.de

Medien-
Technik

XML Namespace

Verschiedene Dokumente können gleichlautende Elementnamen oder Attributnamen enthalten, die verschiedene Bedeutung haben.

HTML-Tabelle

```
<table>
 <tr> <td>Apples</td>
<td>Bananas</td> </tr>
</table>
```

XML-Möbeldaten

```
<table>
 <name>African Coffee Table</name>
 <width>80</width>
<length>120</length>
</table>
```

Medien-
Technik

XML Namespace

Verschiedene Dokumente können gleichlautende Elementnamen oder Attributnamen enthalten, die verschiedene Bedeutung haben.

HTML-Tabelle

```
<h:table>
 <h:tr> <h:td>Apples</h:td>
<h:td>Bananas</h:td> </h:tr>
</h:table>
```

XML-Möbeldaten

```
<´f:table>
 <f:name>African Coffee Table</f:name>
 <f:width>80</f:width>
<f:length>120</f:length>
</f:table>
```

Medien-
Technik

XML Namespace

Verschiedene Dokumente können gleichl
Elementnamen oder Attributnamen en
die verschiedene Bedeutung haben.

Eindeutige URL als Namensraum, dort muss nichts stehen, muss nicht existieren

HTML-Tabelle

```
<table xmlns:www="http://www.w3.org/TR/html4/">
 <www:tr> <www:td>Apples</www:td>
<www:td>Bananas</www:td> </www:tr>
</www:table>
```

XML-Möbeldaten

```
<table xmlns="http://www.ikea.se/Sommerkatalog/">
 <name>African Coffee Table</name>
 <width>80</width> <length>120</length>
</table>
```

Medien-
Technik

XML Schema

- Ersatz für DTD
- Definiert die Elemente eines Dokumentes
- Definiert deren Attribute
- Definiert die Unterlemente, ihre Reihenfolge und Anzahl
- Definiert, ob ein Element leer ist oder Text enthält
- Definiert den Datentyp von Elementen und Attributen
- Definiert feste Werte oder Vorgabewerte für Elemente oder Attribute
- XML Schemata sind in XML geschrieben
  xmlns:xs="http://www.w3.org/2001/XMLSchema „“

Medien-
Technik

XML Schema Beispiel

Note.xsd

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" ...>
    <xs:element name="note„>
        <xs:complexType>
            <xs:sequen*ce>
                <xs:element name="to" type="xs:string"/>
                <xs:element name="from" type="xs:string"/>
                <xs:element name="heading" type="xs:string"/>
                <xs:element name="body" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

Medien-
Technik

XML-Datei mit Schema verknüpfen

```
<?xml version="1.0"?>
<note xmlns="http://www.w3schools.com"
    xmlns:xsi="http://www.w3.org/2001/XMLSche
    ma-instance" xsi:schemaLocation="http://.../
    note.xsd">
<to>Tove</to>
<from>Jani</from>
        <heading>Reminder</heading>
        <body>Don't forget me this
    weekend!</body>
</note>
```

Med
Tech

XML Schema (XSD)

http://www.w3.org/XML/Schema:

"XML Schemas express shared vocabularies and allow machines to carry out rules made by people. ***They provide a means for defining the structure, content*** *and semantics* ***of XML documents****. […]* XML Schema was approved as a W3C Recommendation on 2 May 2001."

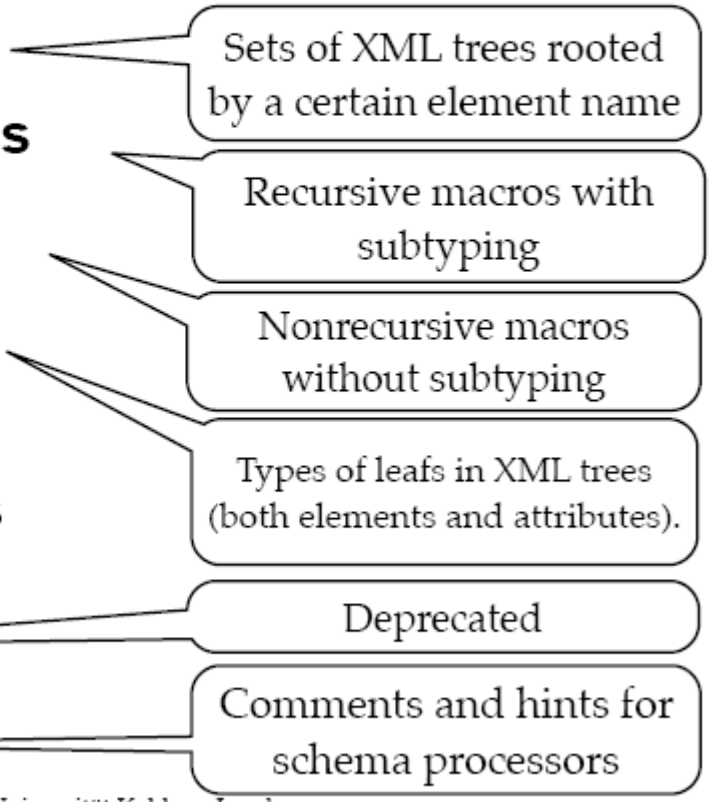
Med
Tech

Unofficial description of XSD

- XSD is an EBNF-like formalism for XML (not for text).
- Even among industry standards, it's particularly ugly.
- It illustrates the "tradeoffs" of large standardization groups.
- XSD takes the XML mess to a whole, new level.
- It violates various principles of language design.
- It is (was) fun thinking about / working on XSD.
- XSD is there to stay for some time (cf. Cobol and Java).
- Not so much Cobol but rather XSD cripples the mind.

XSD – schema components

- **Element declarations**
- **Complex type definitions**
- Model-group definitions
- Simple type definitions
- Attribute declarations
- Attribute-group definitions
- Redefinitions
- Annotations

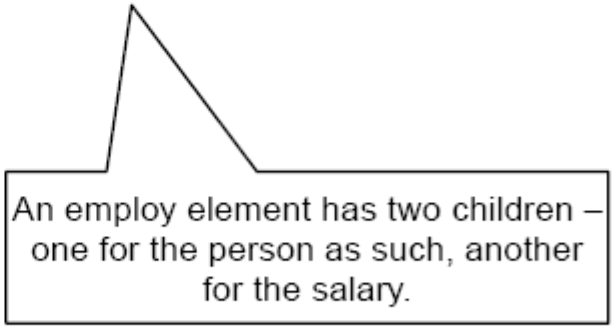


Model group compositors

- <sequence> … juxtaposition in EBNF
- <choice> … “|” in EBNF
- <all> … “||” (permutation phrases)
- minOccurs=“1” maxOccurs=“unbounded” … +
- minOccurs=“0” maxOccurs=“unbounded” … *

<sequence>

```
<xs:complexType name="employee">
   <xs:sequence>
        <xs:element ref="person" />
        <xs:element ref="salary" />
   </xs:sequence>
</xs:complexType>
```



<choice>

```
<xs:element name="subunit">
   <xs:complexType>
        <xs:choice>
            <xs:element name="pu" type="employee" />
            <xs:element name="du" type="dept" />
        </xs:choice>
   </xs:complexType>
</xs:element>
```

A subunit (of a department) is either a person unit (“pu”) or a department unit (“du”).

Med
Tech

“ * ”

A company element has any number of department elements as its children.

```
<xs:element name="company">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="dept"
                type="dept"
                minOccurs="0"
                maxOccurs="unbounded" />
        </xs:sequence>
    </xs:complexType>
</xs:element>
```

Medien-
Technik

XML Schema Datentypen

Definition eines typisierten Elementes:

- <xs:element name="xxx" type="yyy"/>
  wobei yyy einen der Bezeichner
  - xs:string
  - xs:decimal
  - xs:integer
  - xs:boolean
  - xs:date
  - xs:time

Definition von Atributen analog

- <xs:attribute name="xxx" type="yyy"/>
- <xs:attribute name="lang" type="xs:string" use="required"/>

Medien-
Technik

XML Browseransicht

Mi t XML ist keine Druckformat verknüpft:
Browser-Default: Syntay Highlighting

CD_Catalog.xml

CD_Catalog_with_Css.xml

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!-- Edited with XML Spy v4.2  -->
- <CATALOG>
- <CD>
  <TITLE>Empire Burlesque</TITLE>
  <ARTIST>Bob Dylan</ARTIST>
  <COUNTRY>USA</COUNTRY>
  <COMPANY>Columbia</COMPANY>
  <PRICE>10.90</PRICE>
  <YEAR>1985</YEAR>
  </CD>
```

Medien-
Technik

XML Browseransicht


cd_catalog_with_css.xml

Formatierung mit Cascading Style Sheets CSS

Empire Burlesque Bob Dylan
USA
Columbia
10.90
1985

Hide your heart Bonnie Tyler
UK
CBS Records
9.90
1988

Greatest Hits Dolly Parton
USA
RCA
9.90
1982

```
CATALOG
{
background-color: #ffffff;
width: 100%;
}
CD
{
display: block;
margin-bottom: 30pt;
margin-left: 0;
}
TITLE
{
color: #FF0000;
font-size: 20pt;
}
ARTIST
{
color: #0000FF;
font-size: 20pt;
}
,,,,,
```

Medien-
Technik

XML Encoding

XML verwendet defaultmäßig Unicode, ansonsten muss die Codierung angegeben werden:

```
<?xml version="1.0" encoding="ISO-8859-15"?>
```

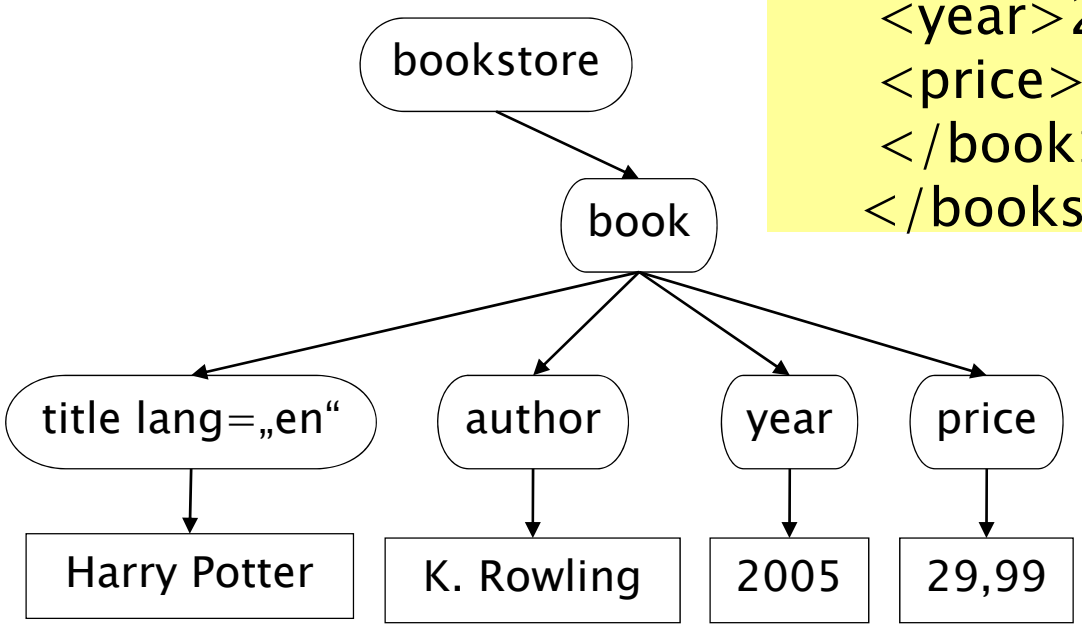
```
<?xml version="1.0" encoding=„UTF-8"?>
```

```
<?xml version="1.0" encoding=„UTF-16"?>
```

Medien-
Technik

XPath / DOM Document Object Model

```
<?xml version="1.0" encoding="ISO-8859-
    1"?>
    <bookstore>
    <book>
     <title lang="en">Harry Potter</title>
     <author>J K. Rowling</author>
     <year>2005</year>
     <price>29.99</price>
     </book>
    </bookstore>
```



Medien-
Technik

XPath: Pfadausdrücke

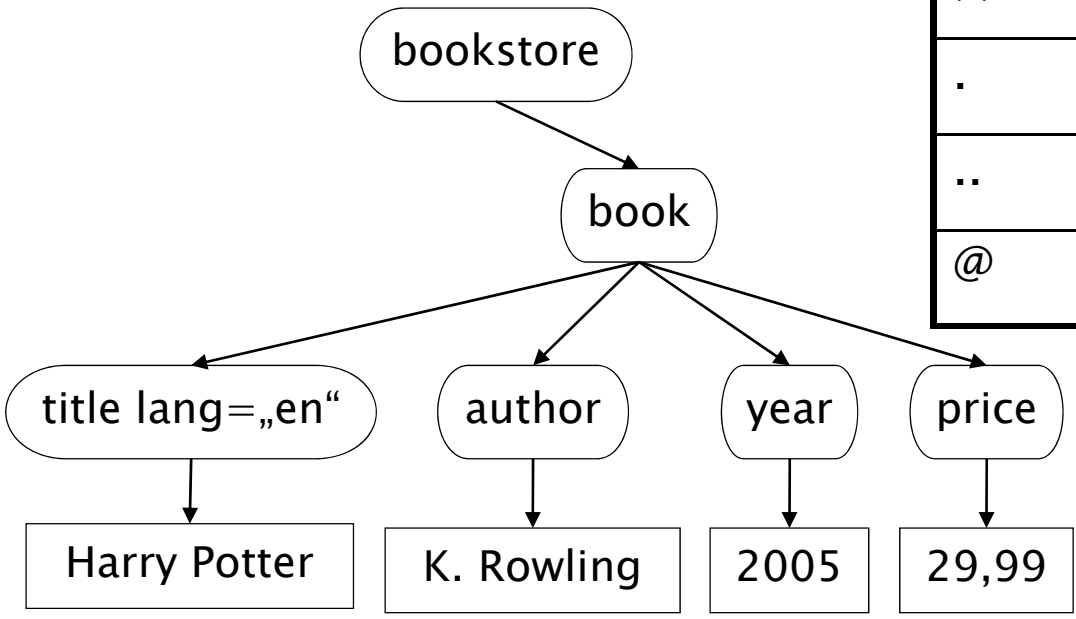
- Die *XML Path Language* (XPath) bietet Sprachmittel zur Beschreibung von Knoteneigenschaften innerhalb der Baumdarstellung eines XML-Dokuments und ermöglicht so eine strukturabhängige Adressierung von Dokumentbestandteilen
- Ein Pfadausdruck dient dazu, aus einer Menge von XPath-Knoten diejenigen Knoten herauszufiltern, die bestimmte Kriterien erfüllen.
- Die Auswertung erfolgt im Kontext der Baumdarstellung eines Quelldokuments, wobei ein Kontextknoten (*Context node*) als Referenzpunkt für bestimmte Prädikate dient.
- Das Ergebnis ist eine Knotenmenge (*Node set*), in der jeder Knoten nur einmal vertreten ist.

http://www.teialehrbuch.de/KCM/16571-Knotenadressierung.html

Medien-
Technik

XPath: Pfadausdrücke

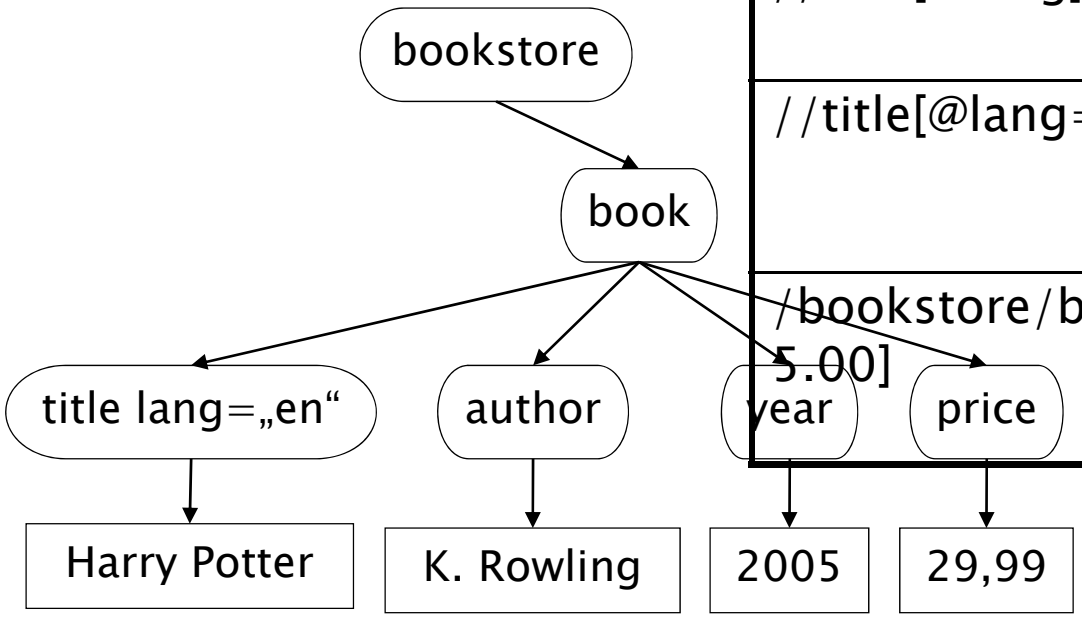
element	Alle Nachfolgeknoten von element
/	Pfad beginnt an der Wurzel
//	Alle Knoten ab dem aktuellen
.	Der aktuelle Knoten
..	Vorgängerknoten
@	Attribut



Medien-
Technik

XPath: Pfadausdrücke

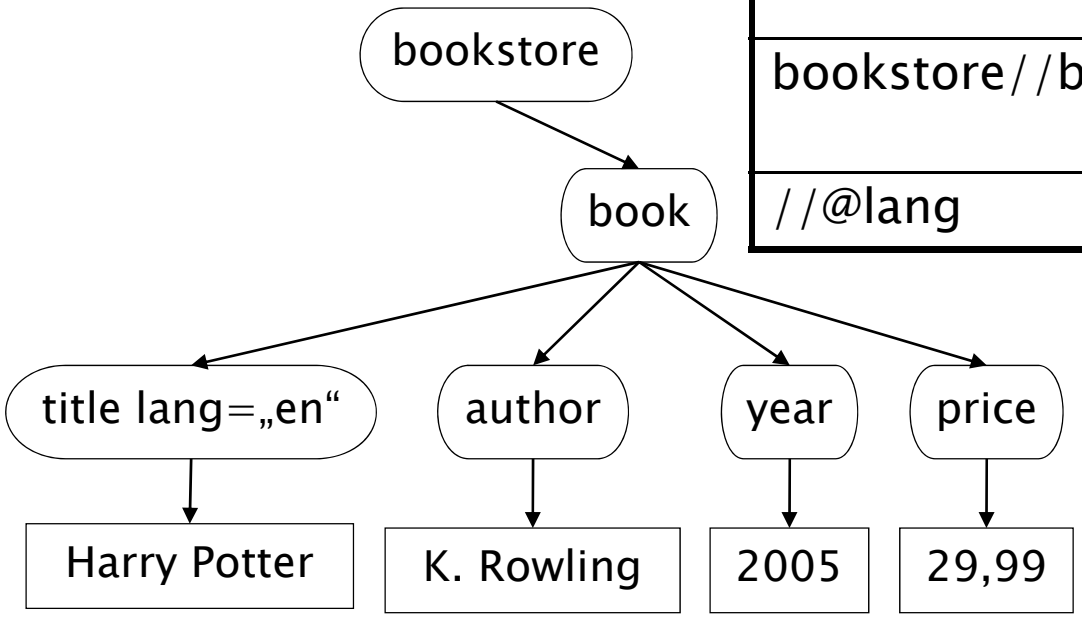
/bookstore/book[1]	Der 1. Book-Knoten unter bookstore
/bookstore/book[last()-1]	Der vorletzte …..
/bookstore/book[position()<3]	Die ersten beiden
//title[@lang]	Alle title-Knoten mit einem lang-Attribut
//title[@lang='eng']	Alle title-Knoten mit einem lang-Attribut, dessen Wert „eng“ ist.
/bookstore/book[price>35.00]	Alle book-Knoten mit einem price-knoten mit Inhalt >35.00



Medien-
Technik

XPath: Pfadausdrücke

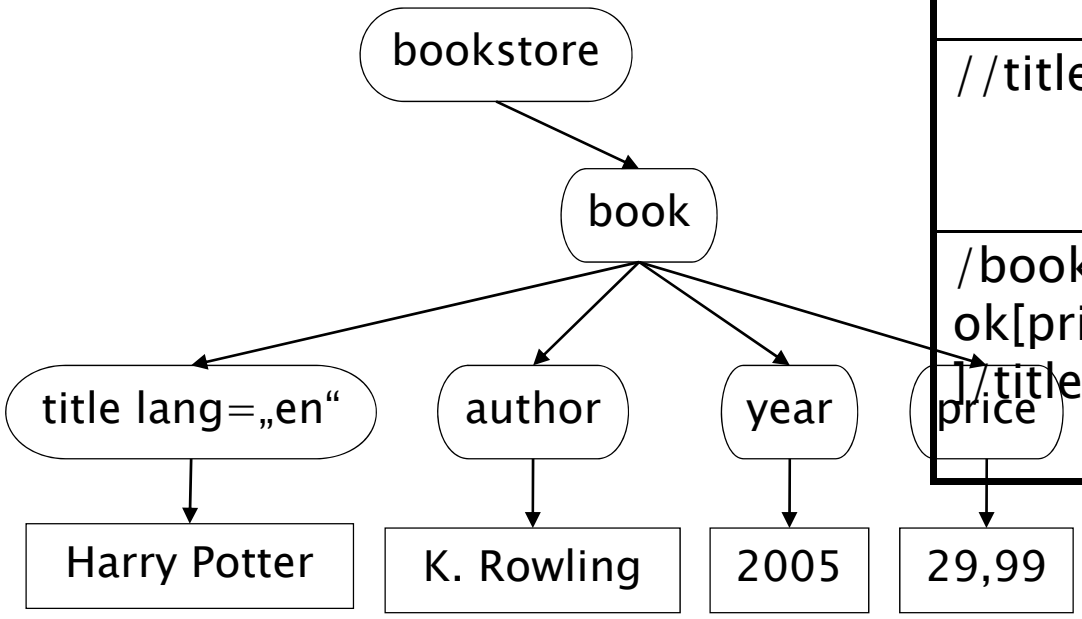
bookstore	book
/bookstore	Wurzelknoten bookstore
bookstore/book	Alle book-Knoten die Kinder von bookstore sind
//book	Alle book-Knoten im Dokument, egal wo
bookstore//book	Alle book-Knoten, die unterhalb von bookstore vorkommen
//@lang	Alle Attribute mit Namen lang



Medien-
Technik

XPath: Pfadausdrücke

*	wildcard
bookstore/*	alle Kinder von bookstore
//*	Alle Elemente im Dokument
//book	Alle book-Knoten im Dokument, egal wo
//title[@*]	alle title-Knoten mit keinem irgendeinem Attribut
/bookstore/book[price>35.00]/title	Alle title-Knoten, die Nachfolger von book-Knoten mit price-Nachfolger-Wert > 35.00



Medien-
Technik

Java und XML

Medien-
Technik

Java & XML

Zugriff auf XML-Dokumente über **DOM** Document Object Modell

The Document Object Model is a platform- and language-neutral interface that will allow programs and scripts to dynamically access and update the content, structure and style of documents. The document can be further processed and the results of that processing can be incorporated back into the presented page. This is an overview of DOM-related materials here at W3C and around the web.

Medien-
Technik

XML verarbeiten mit Java
XML DOM SAX

```
mport org.w3c.dom.*;

import javax.xml.parsers.*;
import javax.xml.transform.*;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;

//Create instance of DocumentBuilderFactory
    DocumentBuilderFactory factory =
   DocumentBuilderFactory.newInstance();
    //Get the DocumentBuilder
    DocumentBuilder docBuilder = factory.newDocumentBuilder();
    //Create blank DOM Document
        Document doc = docBuilder.newDocument();
```

<?xml version=„1.0“ …

http://www.roseindia.net/xml/dom/createdomchildelement.shtml

Medien-
Technik

XML verarbeiten mit Java
XML DOM SAX

```
//create the root element
      Element root = doc.createElement("root");
   //append it to the xml tree
      doc.appendChild(root);

     //create a comment
     Comment comment = doc.createComment("This is comment");
     //add in the root element
     root.appendChild(comment);
```

<?xml version="1.0"...>
<root></root>

<?xml version="1.0" .....>
<root><!__This is a comment--><root>

http://www.roseindia.net/xml/dom/createdomchildelement.shtml

Medien-
Technik

XML verarbeiten mit Java
XML DOM

```
//create child element
   Element childElement = doc.createElement("Child");
   //Add the atribute to the child
   childElement.setAttribute("attribute1","The value of Attribute 1");
   root.appendChild(childElement);
```

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<root><!--This is comment--><Child attribute1="The value of Attribute 1"/></root>
```


http://www.roseindia.net/xml/dom/createdomchildelement.shtml

Medien-
Technik

XML Dokument lesen

Demo AccessXmlFile.java

Weitere XML-Anwendungen

- SVG Scalable Vector Graphics
  - Vektorgrafik
- Mathematical Markup Language (MathML) Version 2.0
  - Formel-Aufbau
- CML Chemical Markup Language
  - Molekülstrukturen, Chemische Formeln
- Resource Description Framework (RDF)
  - Resourcen beschreiben durch Subjekt, Prädikat, Objekt
- RSS Really Simple Syndication (Newsfeed)
  - Automatisierte Nachrichtenverbreitung
- XSL eXtensible Stylesheet Language
  - Layout für XML
- XSLT XSL-Transformation
  - Format-Transformationen von XML-Objekten
- XSL-FO XSL-Formatting Objects
- InkML Ink Markup Language
- Web Ontology Language (OWL)
  - Wissensrepräsentation
- SMIL Synchronized Multimedia Language

Medien-
Technik

Microsoft XAML eXtensible Application Markup Language

- XAMl ist XML.
- XAML ist Teil von .NET 3.0 und WPF (windows Presentation Foundation)
- XAML beschreibt das GUI
- XAML ist deklarativ (Man programmiert WAS man haben will, nicht WIE es erzeugt wird.
- XAML-Elemte sind Fenster, Buttons, Panels etc.
- XAML kennt Stylesheets
- XAML-Elemente können mit Code verbunden werden (Ereignisse)
- XAML erlaubt es, GUI-Gestaltung und Programmierung zu trennen.

Medien-
Technik

XAML-Elemente

- Beschreibung eines GUI-Fensters
  - <window> das Fenster einer Windows-Anwendung
  - <canvas> Container für GUI-Elemente, anordnen nach (x,y)

```
<Window x:Class="XAMLExample.Window1"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
Title="XAMLExample" Height="63" Width="299" >
 <Canvas>
  <Button x:Name="btnTest">Test Button </Button>
 </Canvas>
</Window>
```

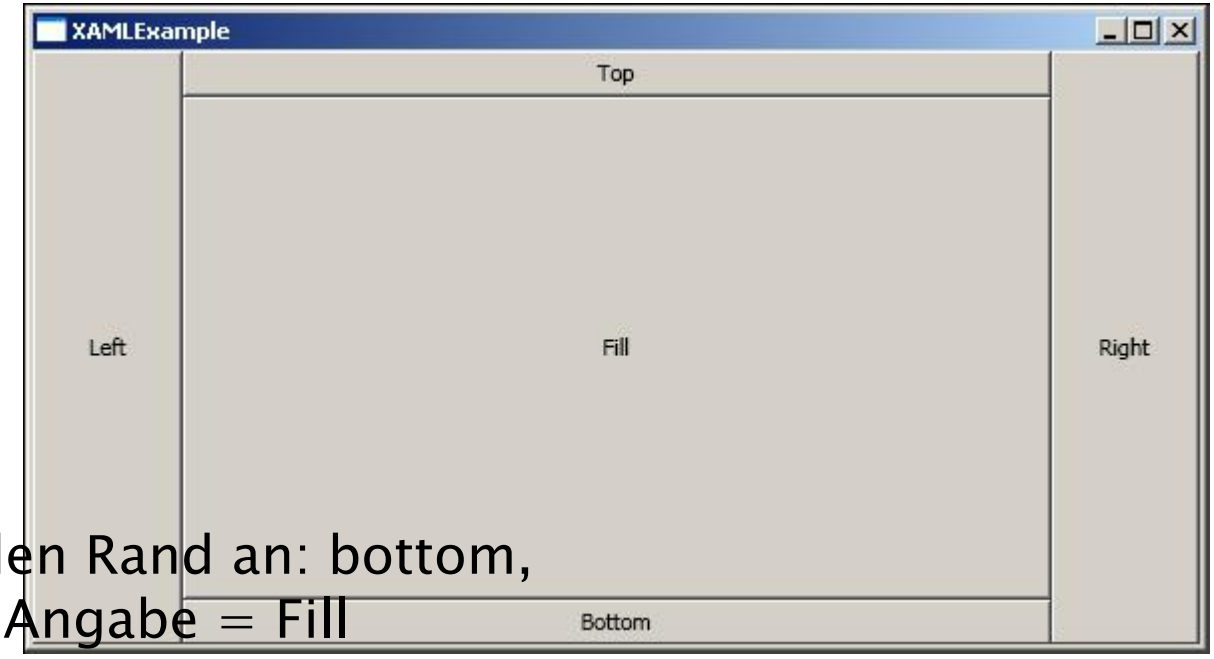
Medien-
Technik

XAML-Elemente

- Weitere Container
  - DockPanel
    - Controls docken an den Rand an: bottom, top, left, right, keine Angabe = Fill
  - StackPanel

```
<Window … Height="63" Width="299" >
  <DockPanel>
    <Button DockPanel.Dock="Top">Top</Button>
    <Button DockPanel.Dock="Bottom">Bottom</Button>
    <Button DockPanel.Dock="Left">Left</Button>
    <Button DockPanel.Dock="Right">Right</Button>
    <Button>Fill</Button>
  </DockPanel>
 </Window>
```

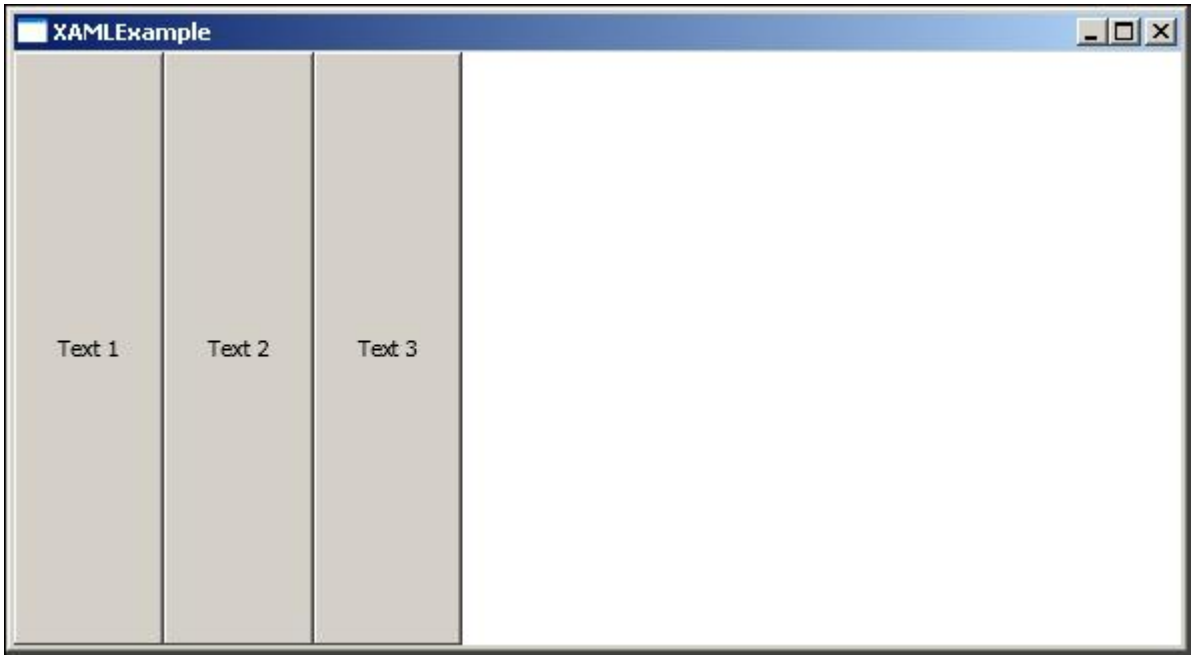
Medien-
Technik



- Weitere Container
  - DockPanel
    - Controls docken an den Rand an: bottom, top, left, right, keine Angabe = Fill

```
<Window … Height="63" Width="299" >
  <DockPanel>
    <Button DockPanel.Dock="Top">Top</Button>
    <Button DockPanel.Dock="Bottom">Bottom</Button>
    <Button DockPanel.Dock="Left">Left</Button>
    <Button DockPanel.Dock="Right">Right</Button>
    <Button>Fill</Button>
  </DockPanel>
 </Window>
```

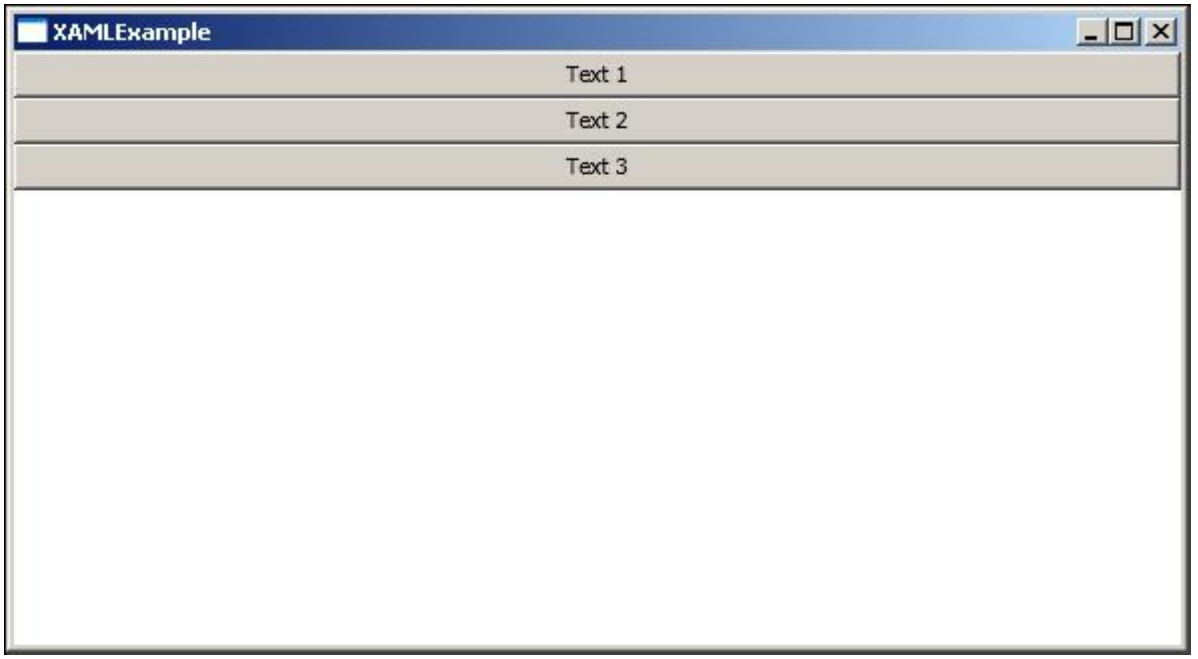
Medien-
Technik



- Weitere Container
  - StackPanel
    - Orientation
      - horizontal

```
<Window … Height="63" Width="299" >
  <StackPanel Orientation="Horizontal">
    <Button>Text 1</Button>
    <Button>Text 2</Button>
    <Button>Text 3</Button>
  </StackPanel>
 </Window>
```

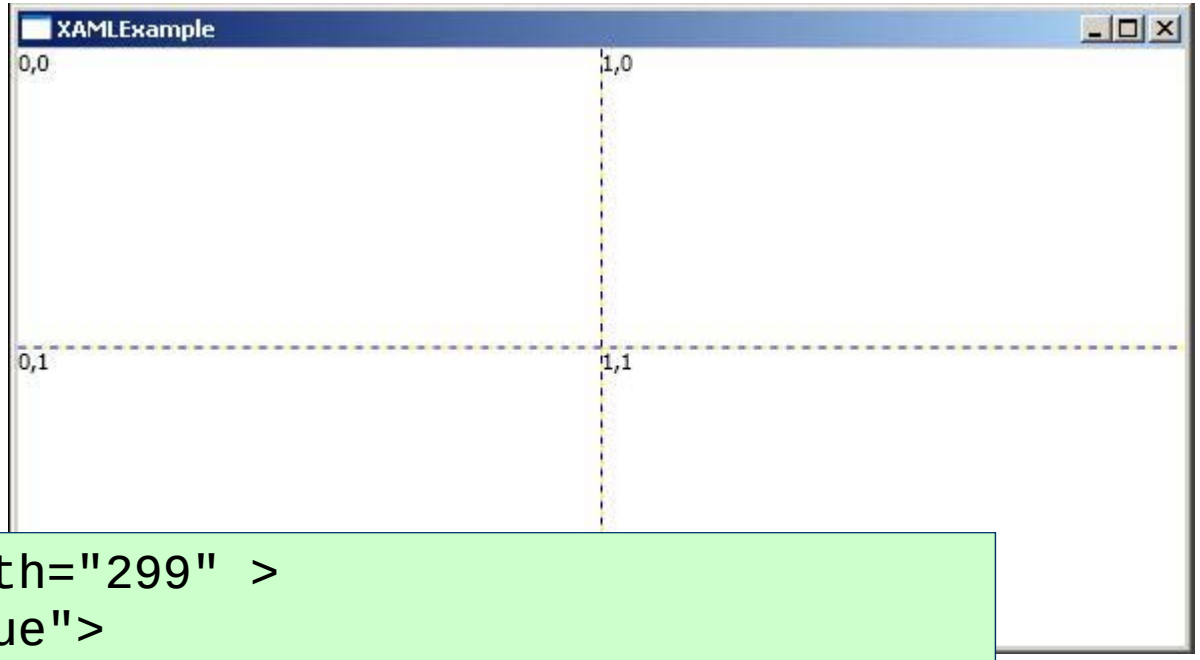
Medien-
Technik



- Weitere Container
  - StackPanel
    - Orientation
      - vertical

```
<Window … Height="63" Width="299" >
  <StackPanel Orientation=„Vertictal">
    <Button>Text 1</Button>
    <Button>Text 2</Button>
    <Button>Text 3</Button>
  </StackPanel>
 </Window>
```

Medien-
Technik



- Weitere Container
  - Grid

```
<Window … Height="63" Width="299" >
  <Grid ShowGridLines="True">
    <Grid.RowDefinitions>
      <RowDefinition /> <RowDefinition />
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
      <ColumnDefinition /> <ColumnDefinition />
    </Grid.ColumnDefinitions>
    <TextBlock Grid.Column="0" Grid.Row="0">0,0</TextBlock>
    <TextBlock Grid.Column="1" Grid.Row="0">1,0</TextBlock>
    <TextBlock Grid.Column="0" Grid.Row="1">0,1</TextBlock>
    <TextBlock Grid.Column="1" Grid.Row="1">1,1</TextBlock>
  </Grid>
</Window>
```

Vorlesung „Medientechnik SS 2010“

Dr. Manfred Jackel
Institut für Informatik
Universität Koblenz-Landau
Postfach 201 602
56072 Koblenz

E-Mail: jkl@uni-koblenz.de
WWW: www.uni-koblenz.de/~jkl
mtech.uni-koblenz.de

Literatur zu diesem Kapitel

Münz, Stefan; Netzger, Wolfgang; HTML & Web-Publishing Handbuch, franzis, Poing, 2002

Norbert Eder: http://blog.norberteder.com/

Ralf Lämmel:

Folien zur Vorlesung „Programierung“ SS 2010

Hyperlinks zu diesem Kapitel

Stefan Münz: SelfHTML

http://www.teamone.de/selfaktuell/

W3C

http://www.w3.org/TR/html40

XML

http://www.w3schools.com
www.xmlfiles.com

http://www.xmltimes.com/

http://www.teialehrbuch.de/KCM/16571-Knotenadressierung.html

Grafik-Quellen

Bild-Logo

www.bild.de