

# Seminar: XML und intelligente Systeme

---

XML-Navigation: XPath, E4X

Christoph Weitkamp  
[cweitkam@techfak.uni-bielefeld.de](mailto:cweitkam@techfak.uni-bielefeld.de)

# Einleitung

---

- XPath ist eine Sprache, um Informationen in XML-Dokumenten zu finden
- Modellierung des XML-Dokuments als Baum
- Navigation durch Elemente und Attribute
- über 100 built-in Funktionen
- Hauptbestandteil in XSLT
- XPath ist ein W3C Standard

# Datenmodell

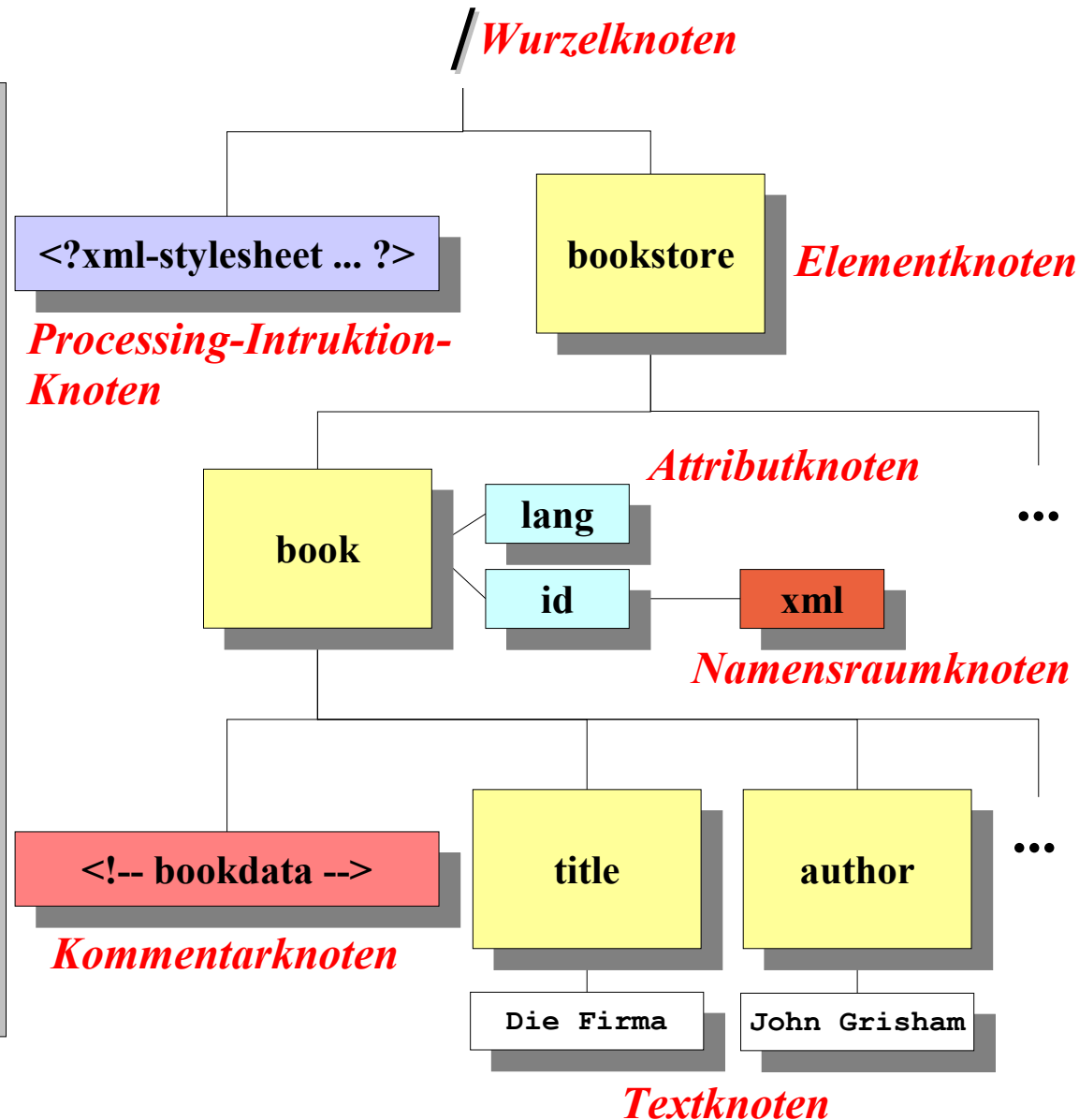
```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```



# Anfragen

---

- gehen meist vom Kontextknoten aus
- Rückgabewert ist Array mit XML Objektknoten, welche die Suchkriterien erfüllen
- oder Rückgabewert der vordefinierten Funktion
  - z.B.: Knotenmengen (node-set), Zeichenketten (strings), Zahlenwerte (numbers) oder boolesche Werte (boolean)

# Grundfunktionen & Variablen

---

- Funktionen auf Knotenmengen
  - z.B.: id, position, last, etc.
- Zeichenkettenfunktionen
  - z.B.: string, translate & andere bekannte Operationen
- Boolesche Funktionen
  - z.B.: boolean, lang, etc.
- Zahlenfunktionen
  - z.B.: number, floor, ceiling, etc.

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

## Knotenauswahl

→ bookstore

bookstore als Kontextknoten

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

## Knotenauswahl

→ /bookstore

bookstore als erster Knoten vom  
Wurzelknoten aus

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

## Knotenauswahl

→ bookstore/book

Alle book Elemente, die Kinder von  
bookstore sind

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

## Knotenauswahl

➔ //book

Alle `book` Elemente, egal wo sie sich im Baum befinden

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

## Knotenauswahl

➔ // @lang

Alle Attribute namens lang

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

## Prädikate

➔ /bookstore/book[1]

Erstes book Element, welches Kind von bookstore ist

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

## Prädikate

→ /bookstore/book[last()]

Letztes book Element, welches Kind  
von bookstore ist

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

## Prädikate

➔ `/bookstore/book[position()<3]`

Alle `book` Elemente, die Kinder von `bookstore` sind und deren `position<3` im Baum ist

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

## Prädikate

➔ //book[@lang='en']

Alle book Elemente, deren Attribut  
lang='en' ist

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

## Prädikate

➔ /bookstore/book[price>20.00]

Alle book Elemente, die Kinder von bookstore sind und deren price>20.00 ist

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

## Prädikate

➔ `/bookstore/book[price<9.99]/title`

title von allen book Elementen, die  
Kinder von bookstore sind und deren  
price<9.99 ist

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

## Unbekannte Knoten

→ /bookstore/\*

Alle Kindknoten von bookstore

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

## Unbekannte Knoten

→ //\*

Alle Elemente im Dokument

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

## Unbekannte Knoten

→ //book[@\*]

Alle `book` Elemente, die ein Attribut haben

# XPath Syntax (Zusammenfassung)

---

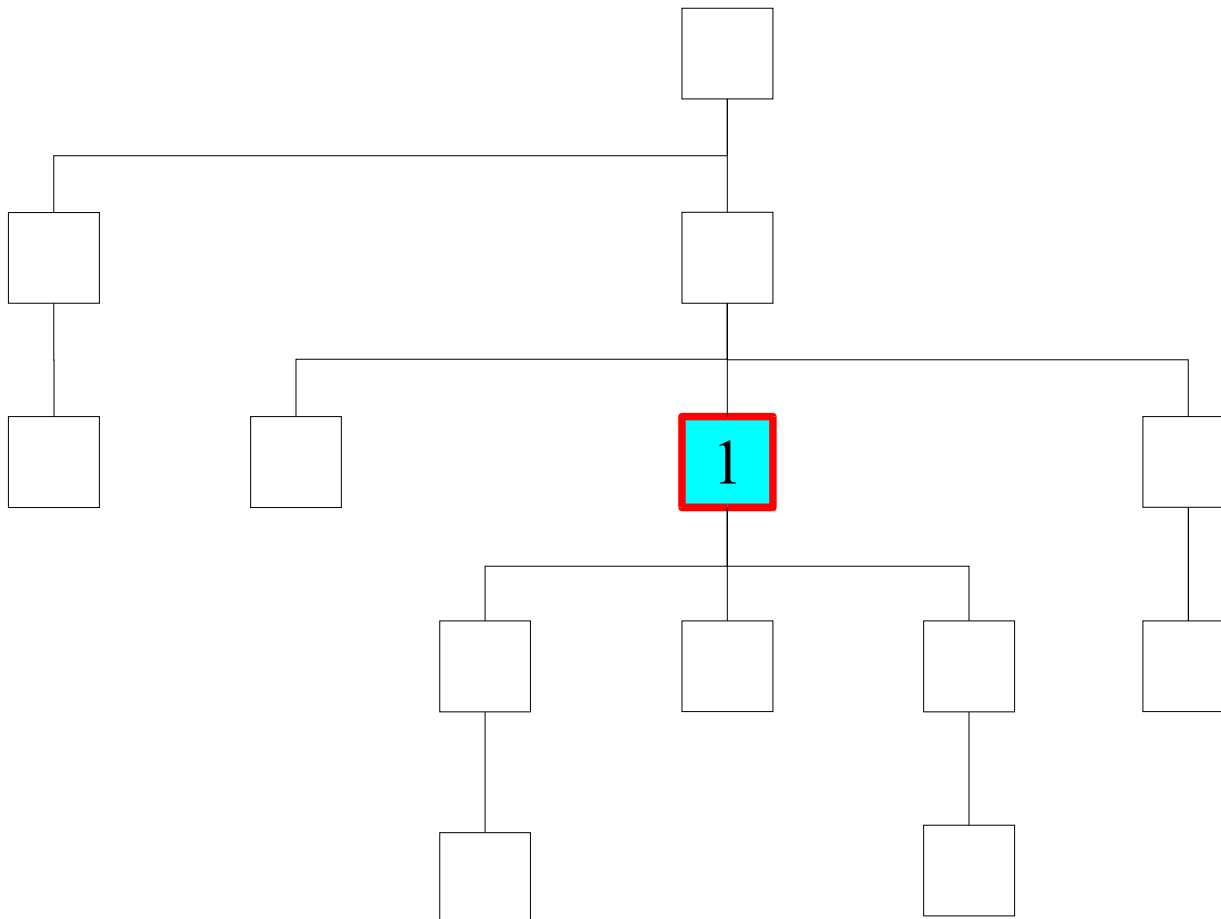
<i>node</i>	Gibt alle Kindknoten von <i>node</i> aus
/	Auswahl der Knoten vom Wurzelknoten aus
// <i>pattern</i>	Auswahl aller Knoten im gesamten Dokument, die auf <i>pattern</i> passen
.	Aktueller Knoten
..	Elternknoten
@	Zugriff auf Attribute

- Prädikate benutzt man, um spezielle Knoten oder Werte zu finden
- Wildcards zur Suche von unbekanntem Knoten

*	Matched auf jeden Elementknoten
@*	Matched auf jeden Attributknoten
node()	Matched auf jeden Knoten jeden Typs

# Achsen

---



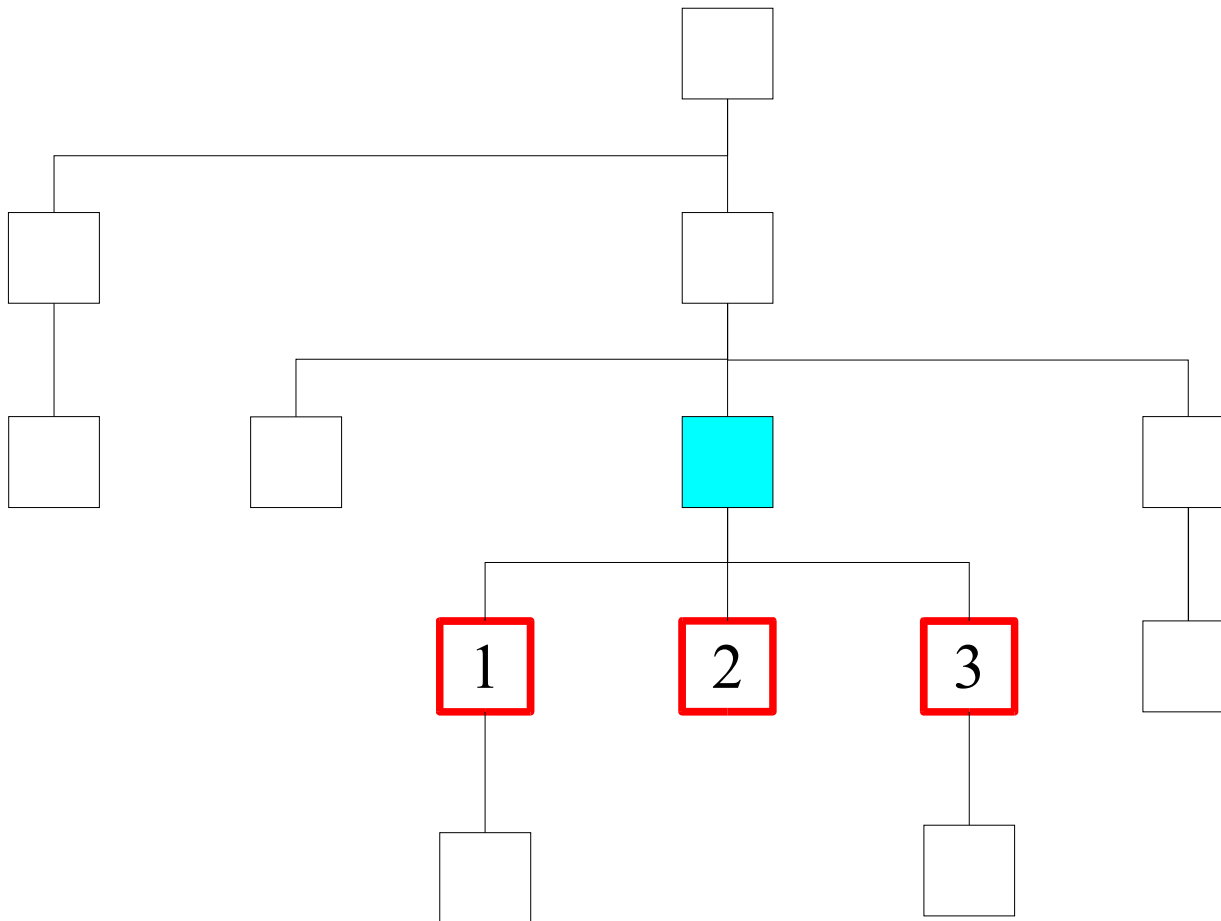
- **self\***
- **child**
- **descendant\***
- **parent**
- **ancestor\***
- **following-sibling**
- **following\***
- **preceding-sibling**
- **preceding\***

■ Kontextknoten

\* Partitionierung des Dokuments

# Achsen

---



 Kontextknoten

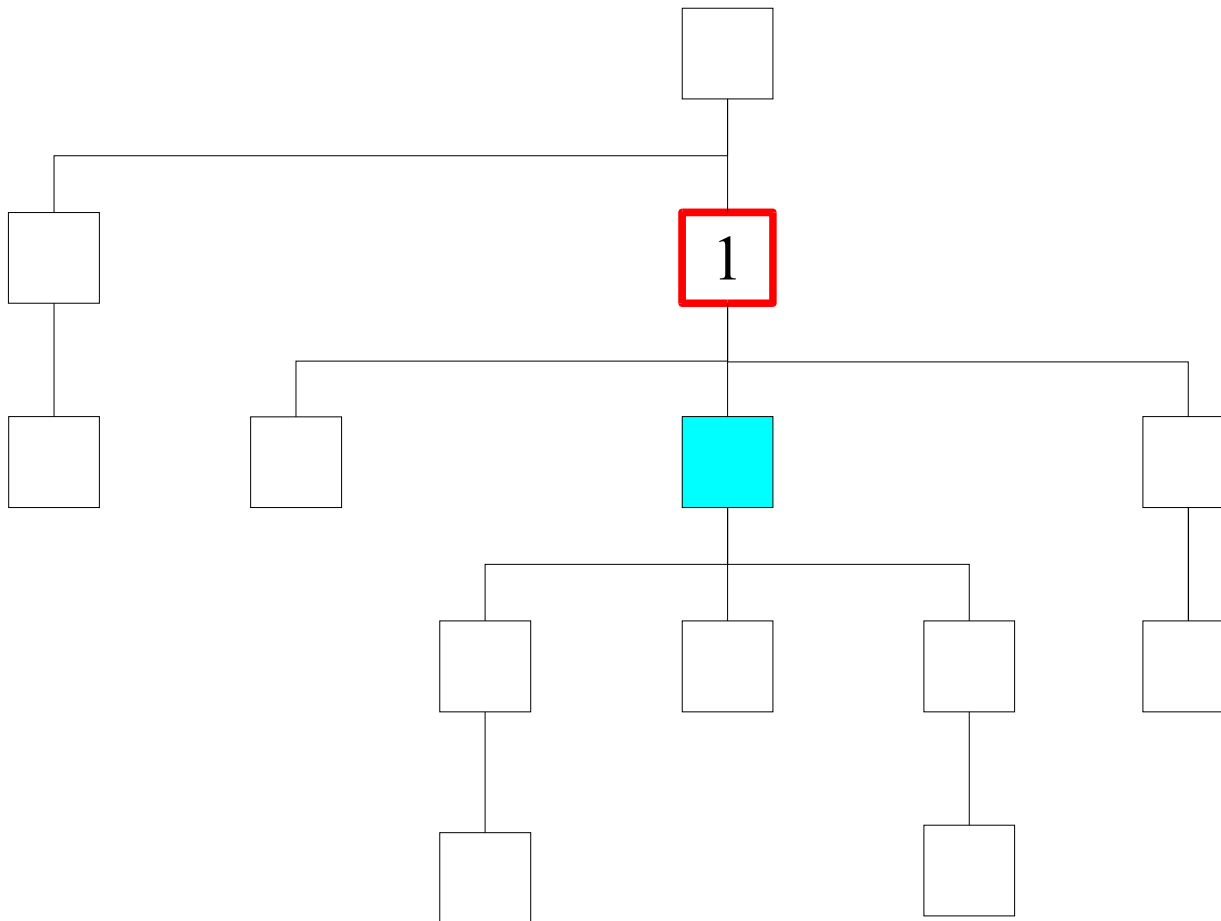
- **self\***
- **child**
- **descendant\***
- **parent**
- **ancestor\***
- **following-sibling**
- **following\***
- **preceding-sibling**
- **preceding\***

\* Partitionierung des Dokuments



# Achsen

---



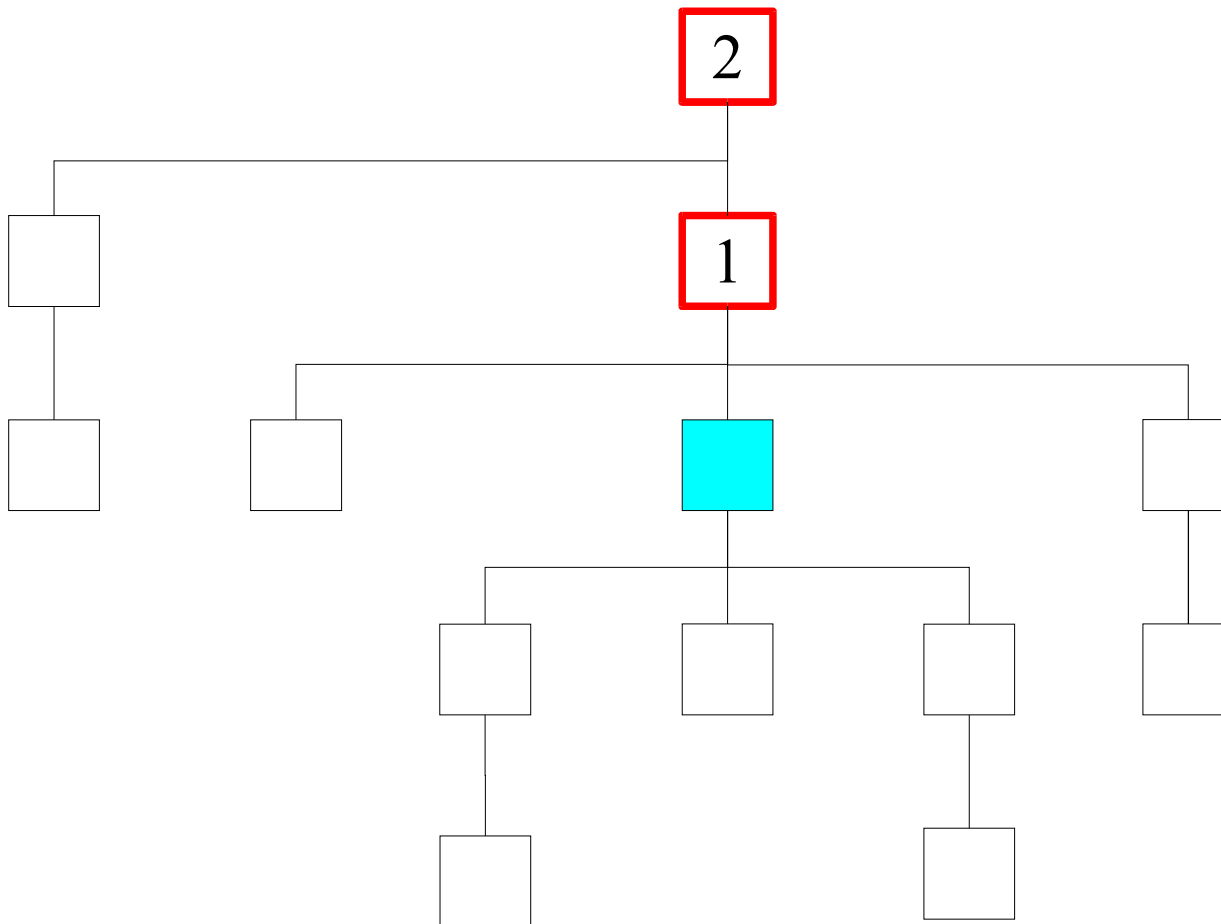
- **self\***
- **child**
- **descendant\***
- **parent**
- **ancestor\***
- **following-sibling**
- **following\***
- **preceding-sibling**
- **preceding\***

 Kontextknoten

\* Partitionierung des Dokuments

# Achsen

---



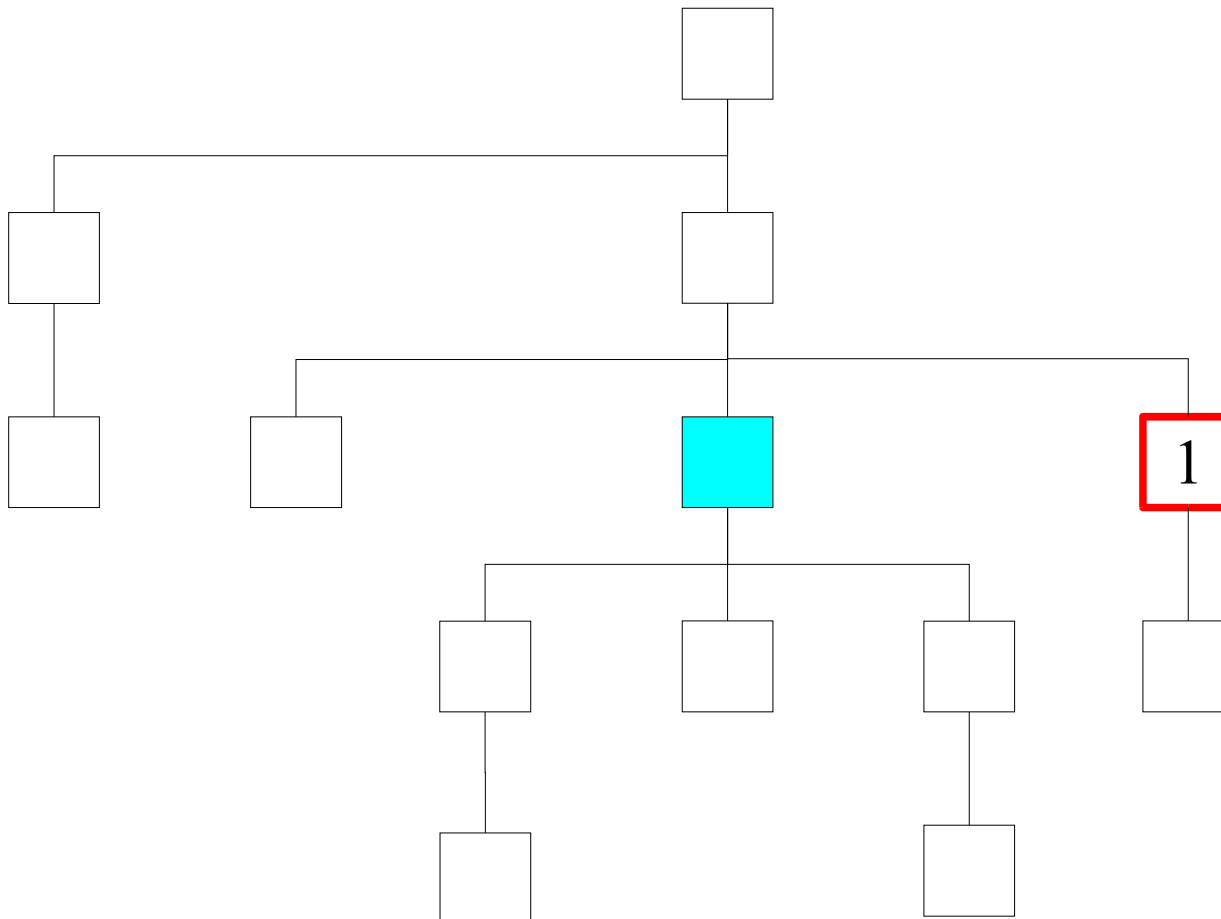
- **self\***
- **child**
- **descendant\***
- **parent**
- **ancestor\***
- **following-sibling**
- **following\***
- **preceding-sibling**
- **preceding\***

 Kontextknoten

\* Partitionierung des Dokuments

# Achsen

---



- **self\***
- **child**
- **descendant\***
- **parent**
- **ancestor\***
- **following-sibling**
- **following\***
- **preceding-sibling**
- **preceding\***

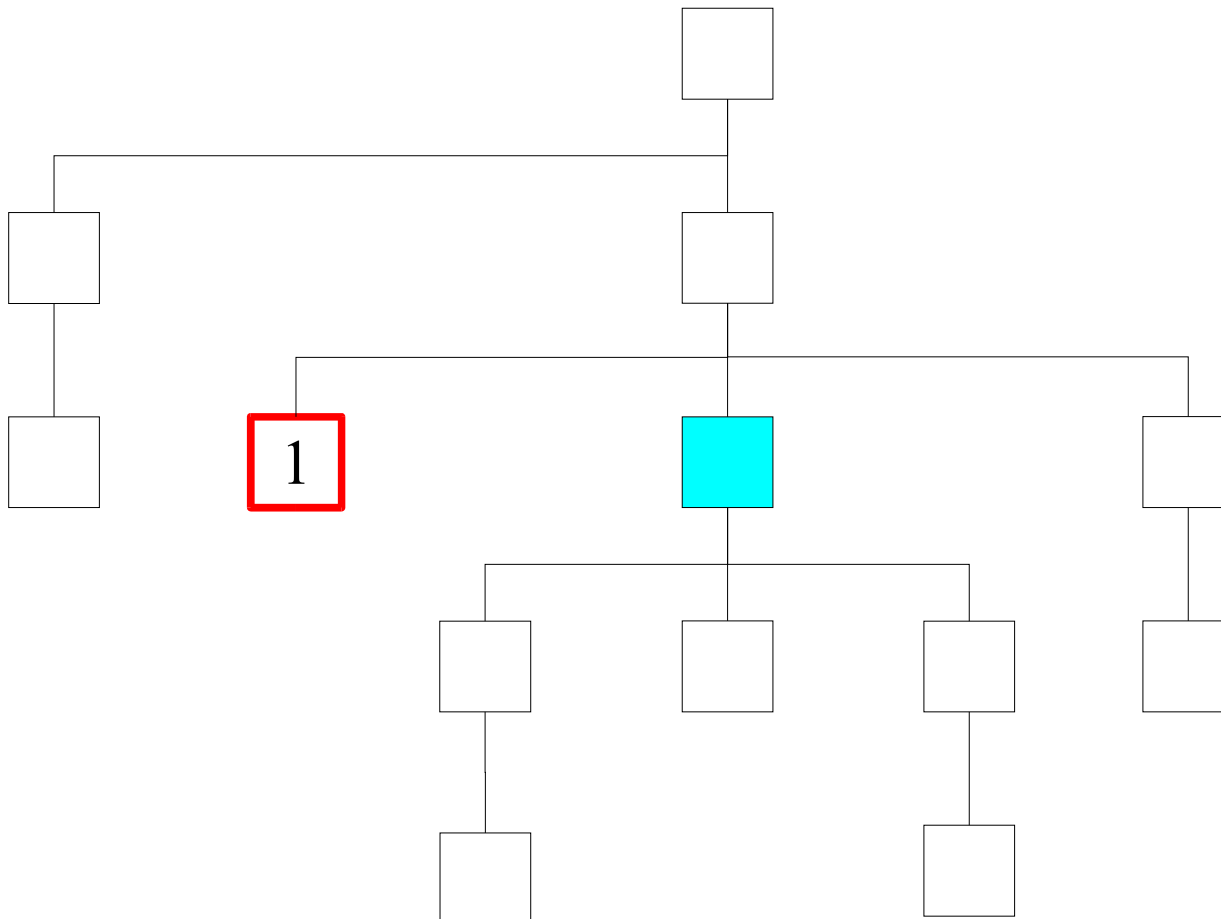
 Kontextknoten

\* Partitionierung des Dokuments



# Achsen

---



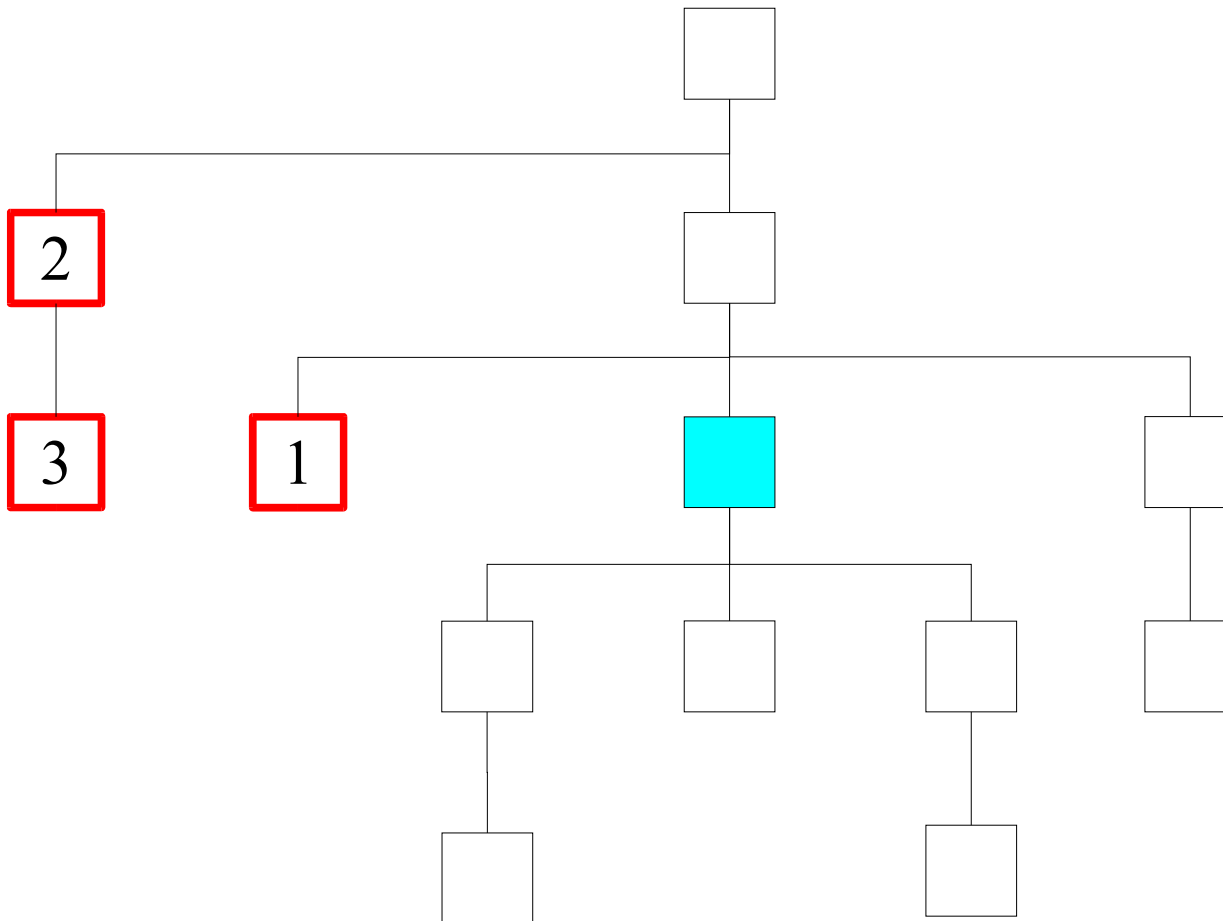
 Kontextknoten

- **self\***
- **child**
- **descendant\***
- **parent**
- **ancestor\***
- **following-sibling**
- **following\***
- **preceding-sibling**
- **preceding\***

\* Partitionierung des Dokuments

# Achsen

---

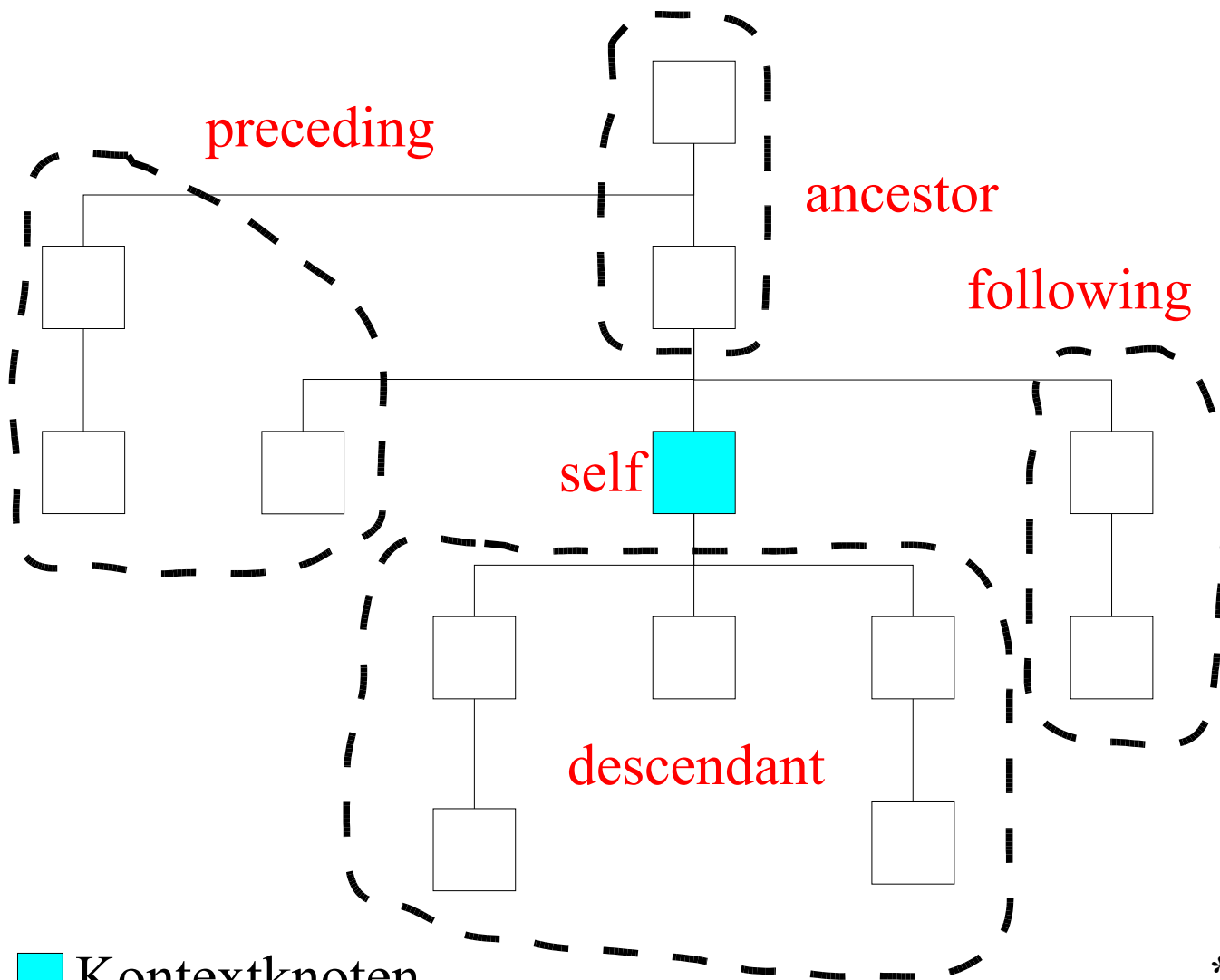


- **self\***
- **child**
- **descendant\***
- **parent**
- **ancestor\***
- **following-sibling**
- **following\***
- **preceding-sibling**
- **preceding\***

 Kontextknoten

\* Partitionierung des Dokuments

# Achsen



- **self\***
- **child**
- **descendant\***
- **parent**
- **ancestor\***
- **following-sibling**
- **following\***
- **preceding-sibling**
- **preceding\***

■ Kontextknoten

\* Partitionierung des Dokuments

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

Achsen - Kontextknoten:

/bookstore

→ child::book

Alle book Knoten, die Kinder von  
bookstore sind

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

## Achsen - Kontextknoten:

/bookstore/book[1]/title

→ ancestor::book

Alle book Knoten, die Vorfahren von  
/bookstore/book[1]/title sind

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

## Achsen - Kontextknoten:

/bookstore/book[2]

→ descendant::\*

## Alle Nachfolgerknoten von

/bookstore/book[2]

# XPath Syntax

---

```
<?xml version="1.0"?>
<?xml-stylesheet href="style.xsl"?>

<bookstore>

<book xml:id="1" lang="en">
  <!-- bookdata -->
  <title>Harry Potte</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29,95</price>
</book>

<book xml:id="2" lang="de">
  <!-- bookdata -->
  <title>Die Firma</title>
  <author>John Grisham</author>
  <year>1996</year>
  <price>4,70</price>
</book>

</bookstore>
```

## Achsen - Kontextknoten:

/bookstore/book[2]/author

→ self::\*

**Der Knoten** /bookstore/book[2]/author  
selbst

# E4X

---

- Erweiterung des ECMAScript mit Unterstützung für XML
- intuitiver und einfacher als XSLT und DOM
- Design Grundlagen:
  - einfach, konsistent, vertraut, minimal,
- komplementär zu XPath
- Benutzbar mit Mozilla Rhino 1.6 (JavaScript)

# E4X Beispiele (1)

---

- zwei Grundlegende neue Datentypen:

- XML-Object:

```
var book = new XML („<book>...</book>“);
```

- XML-List:

```
var booklist = bookstore.book;
```

# E4X Beispiele (2)

---

- Zugriff auf Informationen (intuitiv):

```
var book1 = bookstore.book[0];  
var title1 = book1.title;
```

- Beziehungen:

```
title1.parent();
```

- Attribute:

```
bookstore.book.(@lang="en");
```

- Ausgabe aller Titel:

```
for(i=0; i<bookstore.book.length(); i++) {  
    bookstore.book[i].title.toString();  
}
```

# Diskussion / Aufgaben

---

Ladet die Datei „bookstore.xml“.

Gebt das 4. Buch aus.

Gebt alle die Autoren aller Bücher aus.

Alle Bücher mit Preis == 29,95 ausgeben.

Spielereien!

# E4X Fazit

---

- + Keine neue API lernen
- + schneller Einstieg, geringer Programmieraufwand
- Stand der Implementierung

# Quellen

---

## (1) XML Path Language (XPath)

<http://www.informatik.hu-berlin.de/~obecker/obqo/w3c-trans/xpath-de>

## (2) XPath Tutorial

<http://www.w3schools.com/xpath/default.asp>

## (3) Online XPath Tool

<http://www.activesoftware.com/xml/xpath/>

## (4) ECMAScript for XML (E4X) Specification

<http://www.ecma-international.org/publications/standards/Ecma-357.html>

## (5) Introduction to E4X

<http://www.weblog.infoworld.com/udell/2004/09/29.html>