Freie Universität Berlin

XPath in Depth

The XPath data model

Freie Universität Berlin

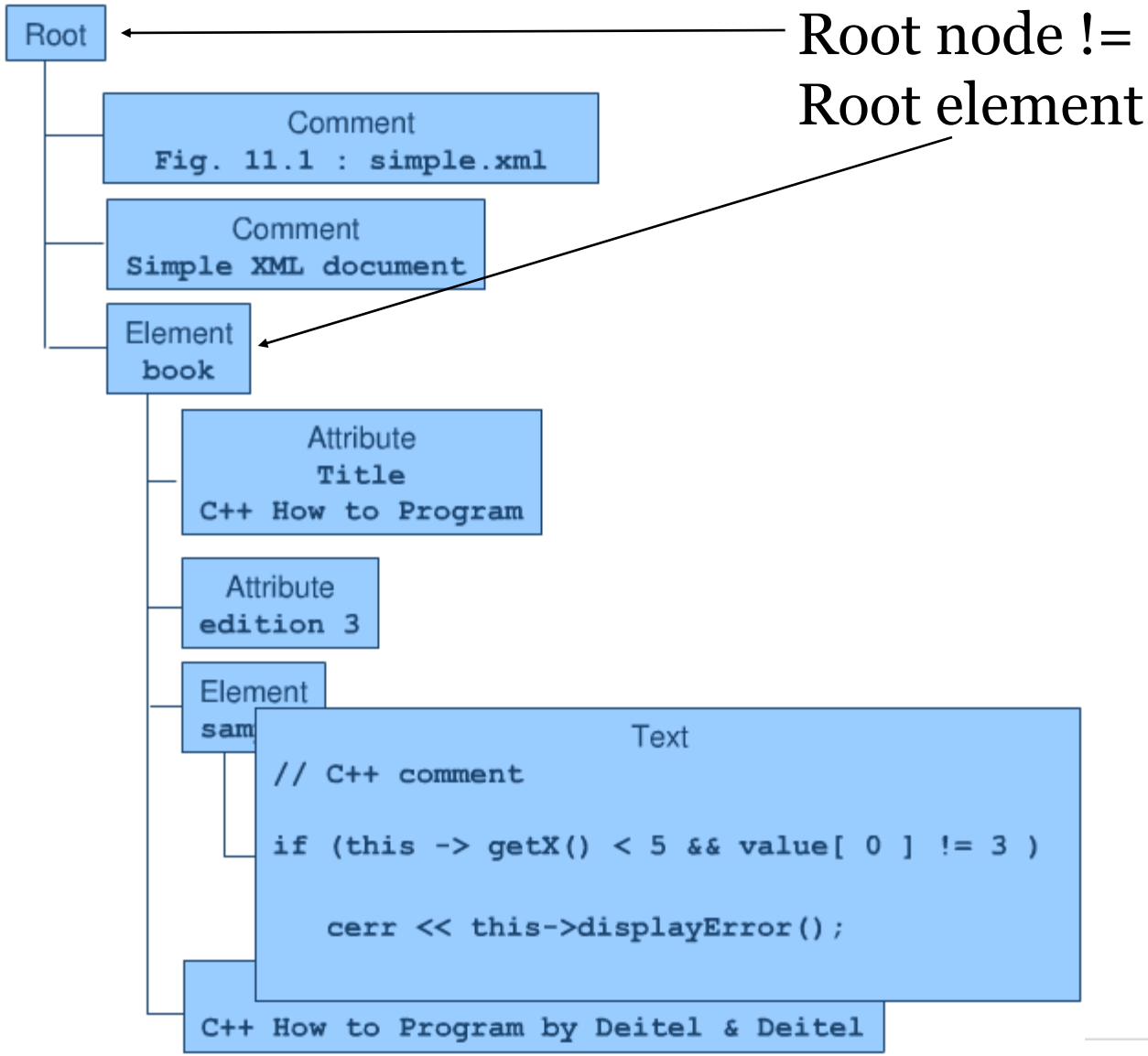
- In XPath, every XML document is a tree of nodes
- There are seven types of node
  - Root (NOT the root element)
  - Element
  - Attribute (NOT a child of the parent node)
  - Text
  - Namespace (NOT a child of the parent node)
  - Processing instruction
  - Comment
- What is NOT part of the XPath tree
  - XML declaration
  - Document type declaration
  - CDATA sections or entity references

XPath data model – XML document

Freie Universität Berlin

```
<?xml version=„1.0“?>
<!-- simple.xml -->
<!-- simple XML document -->
<book title=„C++ How to Program“ edition=„3“>
 <sample>
  <![CDATA[
     //C++ comment
     if ( this->getX() < 5 && value[ 0 ] != 3 )
          oerr << this->displayError();
  ]]>
 </sample>
 C++ How to Program by Deitel &amp;amp; Deitel
</book>
```

XPath data model – XPath tree



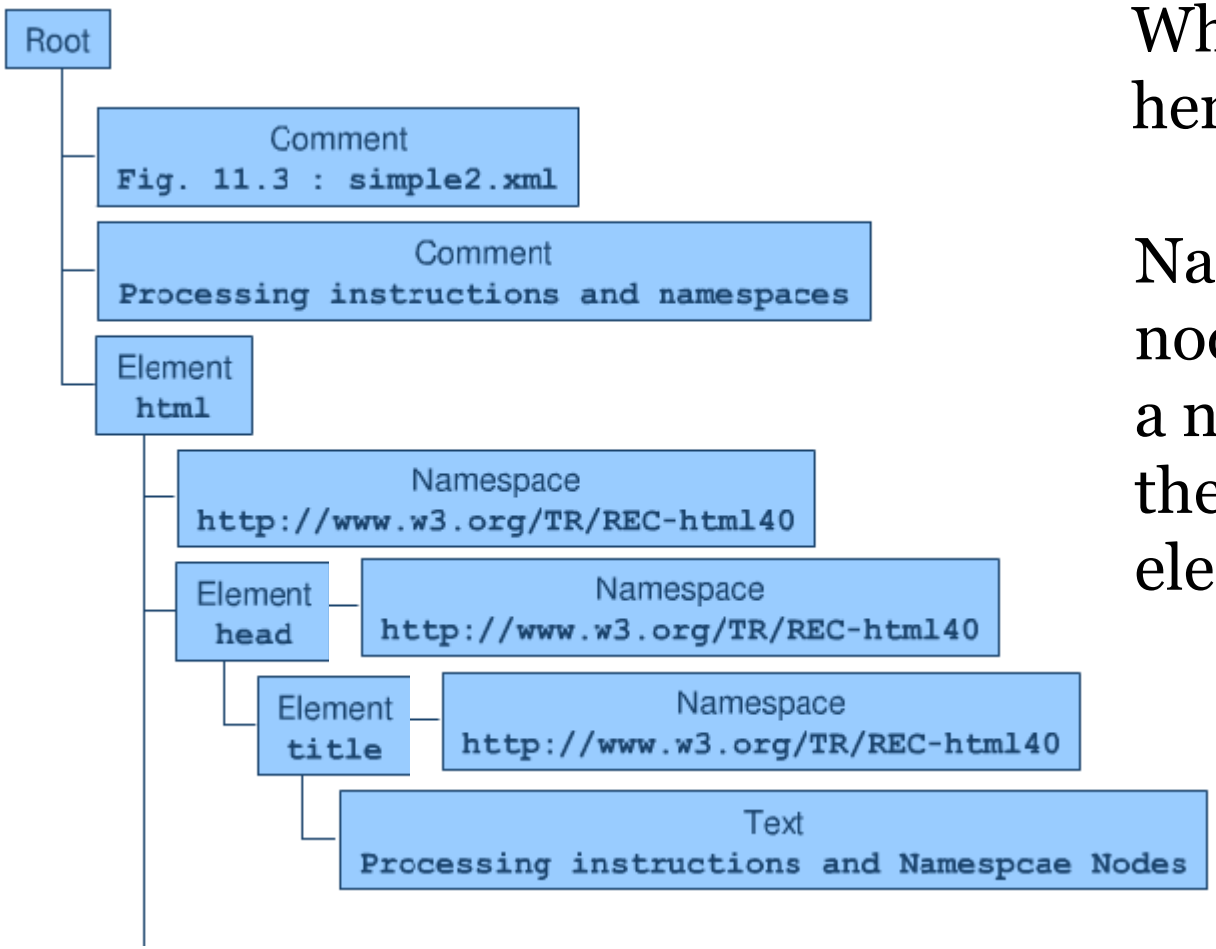
Namespace nodes

Freie Universität Berlin

```
<?xml version=„1.0“?>
<!-- Fig. 11.3 : simple2.xml -->
<!-- Processing instructions and namespaces -->
<html xmlns=„http://www.w3.org/TR/REC-html40“>
 <head>
  <title>Processing Instructions and Namespace
  Nodes</title>
 </head>
…
```

xmlns and xmlns:prefix attributes are NOT attribute nodes

Namespace nodes



What is wrong here?

Namespace nodes represent a namespace in the scope of an element!

XPath node values

Freie Universität Berlin

- Each node has a particular (string) value which it returns if **selected** by a XPath expression
  - *Root node*: the entire text of the XML document
  - *Element node*: the complete, parsed text between the element‘s start and end tags (all tags, comments and processing instructions removed, all CDATA sections and entity references are resolved)
  - *Attribute node*: the normalized attribute value
  - *Text node*: the text content with all CDATA sections and entity references resolved
  - *Namespace node*: the namespace URI
  - *Processing instruction node*: the data of the processing instruction
  - *Comment node*: the text content of the comment

XPath node lists

Freie Universität Berlin

- Often, a XPath expression finds more than one **match** for the context node in the document
- In XPath, this is considered the context node list
- In XSLT, for example, each node in the list will be considered in turn

```
<xsl:template match=„//person“>
  <xsl:value-of select=„//person“ />
</xsl:template>
```

XPath node sets

Freie Universität Berlin

- A node list can be provided to any function which accepts the node-set datatype
- One function id() returns a node-set (of all nodes who have an attribute of type ID which has a value from the input string)

e.g. in XSLT

```
<xsl:template match=„id(„aa ab“)“>
```

Returns a node-set of the elements which have an ID-type attribute with the value aa or ab

```
<xsl:value-of select=„id(„aa ab“)“ />
```

Returns the string values of the elements which have an ID-type attribute with the value aa or ab

XPath predicates

Freie Universität Berlin

- Each location step may have zero or more predicates

/person/profession[.=„doctor“][position()=2]

How to interpret this?

```
<person>
 <profession>doctor</profession>
 <profession>nurse</profession>
 <profession>nurse</profession>
 <profession>doctor</profession>
</person>
```

XPath predicates (2)

Freie Universität Berlin

- XPath resolves predicates from left to right

/person/profession[.=„doctor“][position()=2]

```
<person>
 <profession>doctor</profession>
 <profession>nurse</profession>
 <profession>nurse</profession>
 <profession>doctor</profession>
</person>
```

XPath predicates (3)

Freie Universität Berlin

- XPath resolves predicates from left to right

/person/profession[.=„doctor“][position()=2]

```
<person>
 <profession>doctor</profession>
 <profession>nurse</profession>
 <profession>nurse</profession>
 <profession>doctor</profession>
</person>
```

XPath functions (contd.)

Freie Universität Berlin

- XPath 1.0 has 27 built-in functions
- Others which use XPath, e.g. XSLT or XPointer, extend this function list
- Some XPath/XSLT parsers allow for user-defined extension functions

- More string functions
  - `concat(string 1, string 2…)` returns string
  - `contains(string 1, string 2)` returns boolean
- More number functions
  - `ceiling(number n)` returns smallest whole number > n
  - `floor(number n)` returns largest whole number < n

XPath functions (2)

Freie Universität Berlin

- String manipulation

  - `substring(string s, number index, number length)`
  - `substring(string s, number index)`
  - `substring-after(string s1, string s2)`
  - `substring-before(string s1, string s2)`
  - `translate(string s1, string s2, string s3)`

  e.g. `translate(„I don‘t like the letter l“,“l„,“_“)`

  I don‘t like the letter l → I don‘t _ike the _ette_ _

XPath functions (3)

Freie Universität Berlin

- `lang(string language-code)` returns boolean

The nearest xml:lang attribute on the context node or one of its ancestors determines the language of the node

If no such xml:lang attribute exists, lang() returns false

XPath functions (4)

Freie Universität Berlin

- `name()` returns string
- `name(node-set nodes)` returns string

Returns qualified name (e.g. html:body) of the context node or the first node in the node-set

- `local-name()` returns string
- `local-name(node-set nodes)` returns string

As above, returning only the local name (after the namespace prefix) e.g. for <html:body> returns the string „body“

- `namespace-uri()` returns string
- `namespace-uri(node-set nodes)` returns string

As above, returning only the namespace URI of the node (not the namespace prefix)

XPath functions (5)

Freie Universität Berlin

- Handling whitespace in XML is often necessary, as the XML parser passes normally all whitespace and line breaks into the XML data model without changes
- `normalize-space()` takes a string and normalizes it:
  - stripped of trailing and leading whitespace
  - sequences of whitespace reduced to one whitespace character
  - removes line breaks
- e.g. what is the XML element content for:

  <person> John Edwards </person>

  - Value of `person` is „ John Edwards „
  - Value of `normalize-space(person)` is „John Edwards“

XPath Examples

Beispiele

Freie Universität Berlin

Wähle das Wurzelelement AAA aus:

```
<AAA>
  <BBB/>
  <CCC/>
  <BBB/>
  <BBB/>
  <DDD>
     <BBB/>
  </DDD>
  <CCC/>
</AAA>
```

/AAA

Wähle alle CCC Elemente aus, die Kinder des Elements AAA sind:

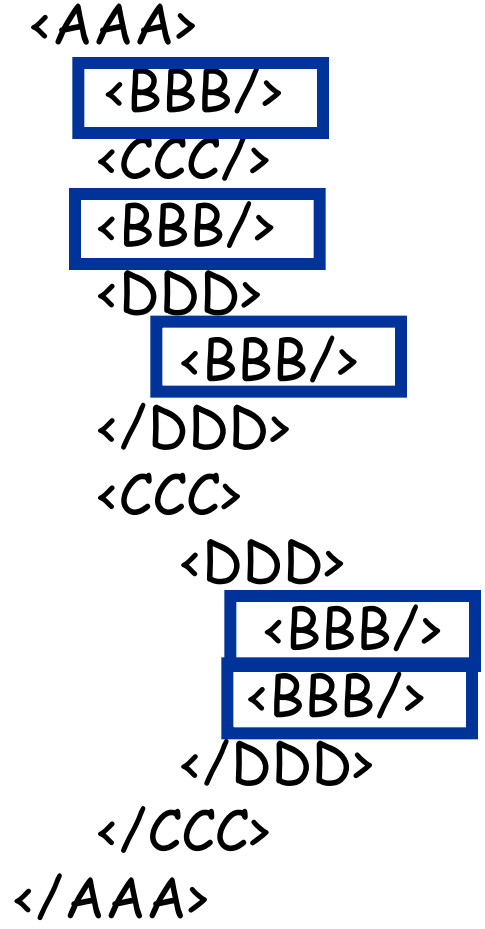
```
<AAA>
    <BBB/>
    <CCC/>
    <BBB/>
    <BBB/>
    <DDD>
       <BBB/>
    </DDD>
    <CCC/>
</AAA>
```

/AAA/CCC

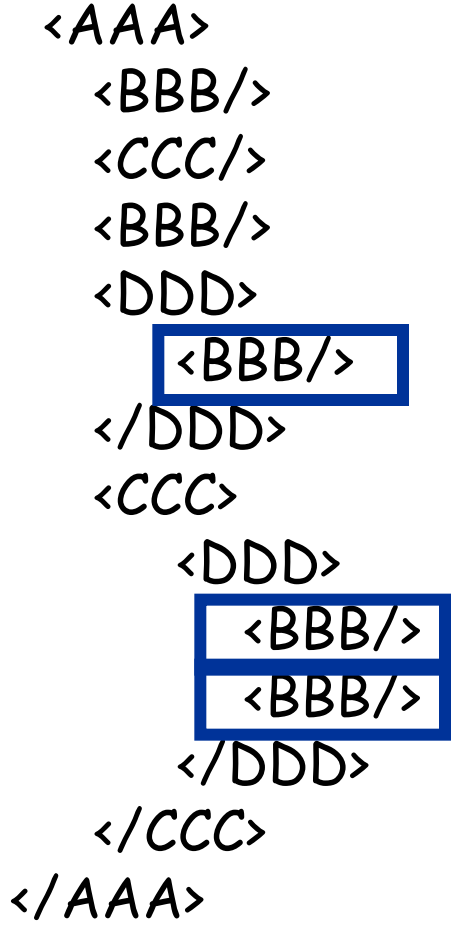
Beispiele

Freie Universität Berlin

//BBB



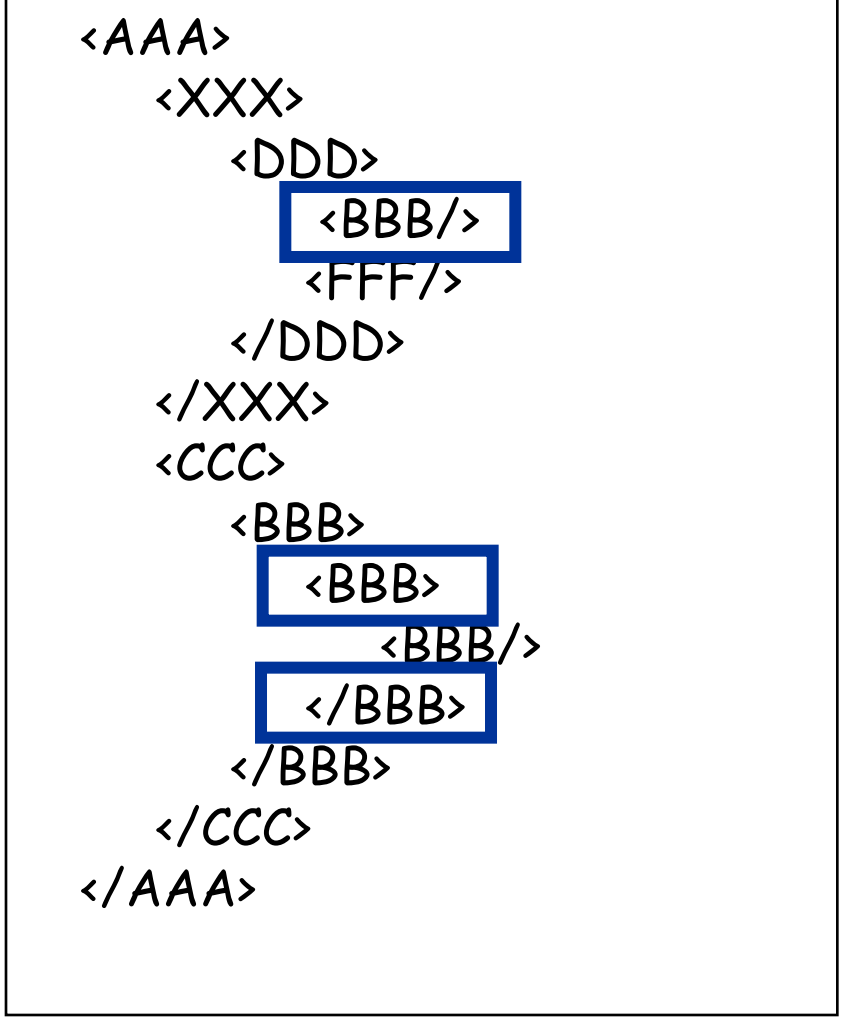
//DDD/BBB



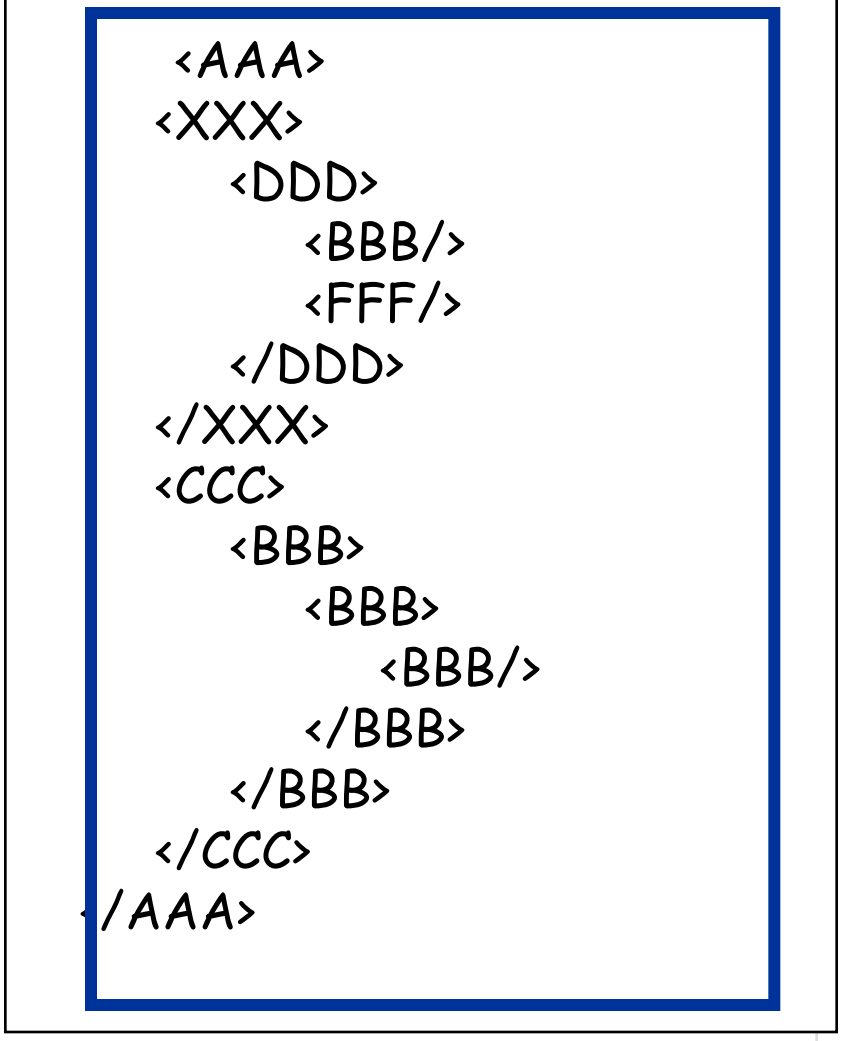
Beispiele

Freie Universität Berlin

/*/*/*/BBB



//*



Beispiele

Freie Universität Berlin

/AAA/BBB[last()]

```
<AAA>
    <BBB/>
    <BBB/>
    <BBB/>
    <BBB/>
</AAA>
```

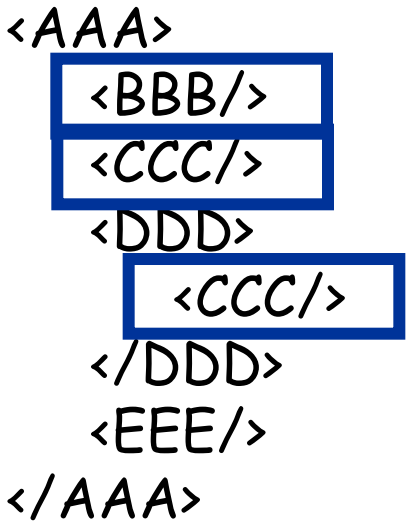
//@id

```
<AAA>
 <BBB id="b1"/>
 <BBB id="b2"/>
  <BBB
        name="bbb"/>
    <BBB/>
</AAA>
```

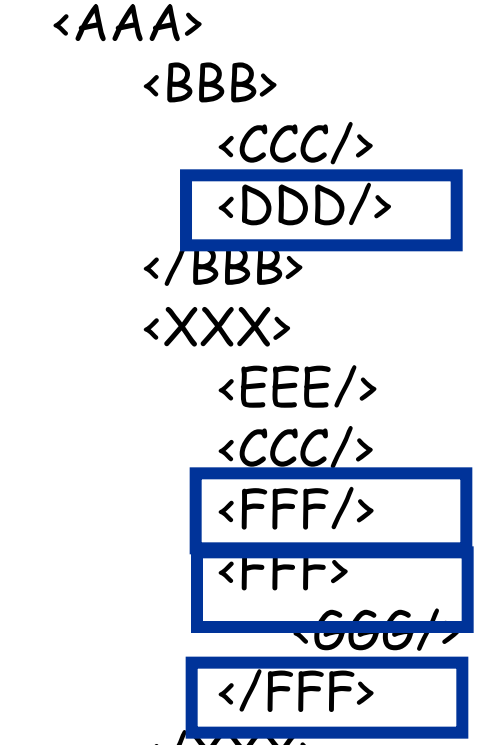
Beispiele

Freie Universität Berlin

//CCC | //BBB



//CCC/following-sibling::*



Musterfragen

Frage 1

Freie Universität Berlin

Which of the following XSLT elements will apply any matching templates to all the children of the current node?

A. <xsl:apply-templates select=„node()“ />

B. <xsl:call-template name=„*“ />

C. <xsl:apply-templates />

D. <xsl:call-template />

E. <xsl:apply-templates name=„*“ />

Frage 2

```
<?xml version="1.0" encoding="UTF-8"?>
<periodicTable>
  <chemicalElement symbol="Ag">
    <atomicNumber>47</atomicNumber>
    <atomicWeight>107.8682</atomicWeight>
  </chemic
</periodicT
```

Which is the output from

<xsl:copy/> ?

A. <chemicalElement>

B. <chemicalElement symbol=„Ag“/>

C. <chemicalElement symbol=„Ag“> …
</chemicalElement>

D. 47 107.8682

Frage 3

```
<?xml version="1.0" encoding="UTF-8"?>
<periodicTable>
  <chemicalElement symbol="Ag">
    <atomicNumber>47</atomicNumber>
    <atomicWeight>107.8682</atomicWeight>
  </chemic
</periodicT
```

Which is the output from

<xsl:copy-of select=„.“ /> ?

A. <chemicalElement>

B. <chemicalElement symbol=„Ag“/>

C. <chemicalElement symbol=„Ag“> …
</chemicalElement>

D. 47 107.8682

Frage 4

```
<?xml version="1.0" encoding="UTF-8"?>
<periodicTable>
  <chemicalElement symbol="Ag">
    <atomicNumber>47</atomicNumber>
    <atomicWeight>107.8682</atomicWeight>
  </chemic
</periodicT
```

Which is the output from

<xsl:value-of select=„.“ /> ?

A. <chemicalElement>

B. <chemicalElement symbol=„Ag“/>

C. <chemicalElement symbol=„Ag“> …
</chemicalElement>

D. 47 107.8682

Frage 5

Freie Universität Berlin

```
<?xml version="1.0"
encoding="UTF-8"?>
<A>
  <B c=„123“>
    <C />
    <D>text</D>
  </B>
  <D>
    <E>more text</E>
    <F c=„246“ />
  </D>
  <F a=„369“ />
</A>
```

```
<xsl:template match=„A“>
  <xsl:apply-templates />
</xsl:template>
<xsl:template match=„B|D|F“>
  <xsl:value-of select=„@*“ />
</xsl:template>
```

Which is the output?

A. Nothing

B. 123 text more text 246 369

C. 123 246 369

D. 123 369

Frage 7

```
<?xml version="1.0"
encoding="UTF-8"?>
<A>
  <B c=„123“>
    <C />
    <D>text</D>
  </B>
  <D>
    <E>more text</E>
    <F c=„246“ />
  </D>
  <F a=„369“ />
</A>
```

```
<xsl:template match=„D“>
  <xsl:apply-templates select=„F“ />
</xsl:template>
<xsl:template match=„F“>
  <xsl:value-of select=„@*“ />
</xsl:template>
```

Which is the output?

A. Nothing

B. text more text 246 369

C. text 246 369

D. 246 369