



# XPath

# XML Path Language

---



- XML-Datei verarbeitet man zum Beispiel
  - durch Transformation mittels XSLT
  - indem man sie mit einem Programm einliest und die Daten dort verwendet
  - im Rahmen eine Ajax Request, wenn ein JavaScript vom Web-Server XML-formatierte Daten geschickt bekommt
- in allen Fällen muss auf die einzelnen Elemente gezielt zugegriffen werden können

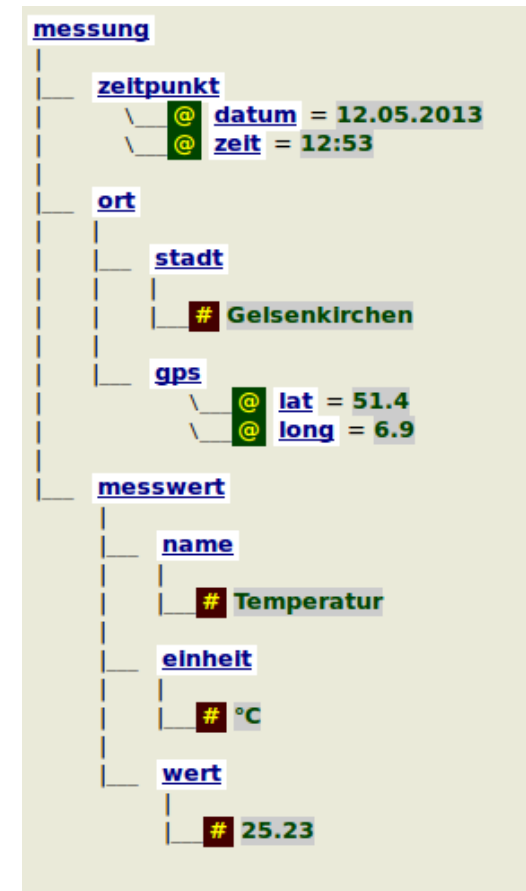
→ **XPath**

# XML → Baum



- Jedes XML-Element ist als Baum darstellbar
- die Bestandteile des Baums sind die „Knoten“ („nodes“)

```
<?xml version='1.0' encoding='UTF-8'?>
<messung>
  <zeitpunkt datum='12.05.2013' zeit='12:53' />
  <ort>
    <stadt>Gelsenkirchen</stadt>
    <gps lat='51.4' long='6.9' />
  </ort>
  <messwert>
    <name>Temperatur</name>
    <einheit>°C</einheit>
    <wert>25.23</wert>
  </messwert>
</messung>
```



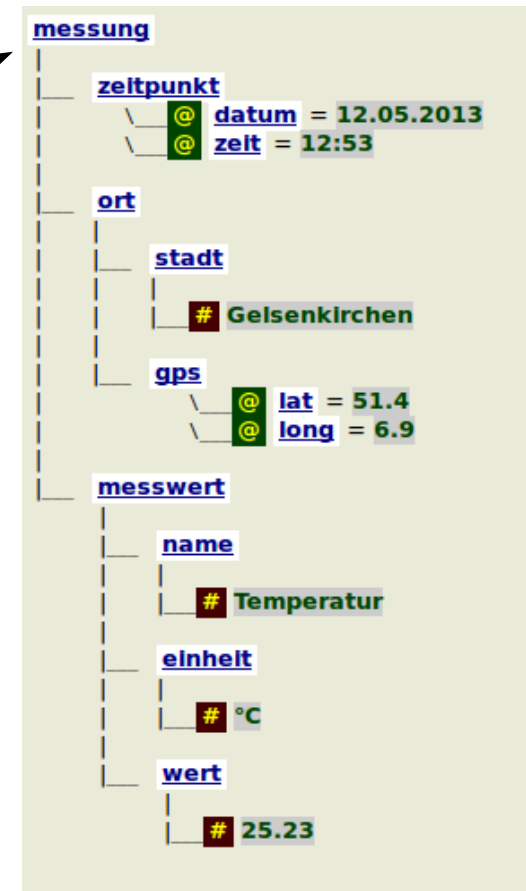
# XML → Baum



- Wurzelknoten: die Wurzel des Baums

```
<?xml version='1.0' encoding='UTF-8'?>
<messung>
  <zeitpunkt datum='12.05.2013' zeit='12:53' />
  <ort>
    <stadt>Gelsenkirchen</stadt>
    <gps lat='51.4' long='6.9' />
  </ort>
  <messwert>
    <name>Temperatur</name>
    <einheit>°C</einheit>
    <wert>25.23</wert>
  </messwert>
</messung>
```

Wurzelknoten



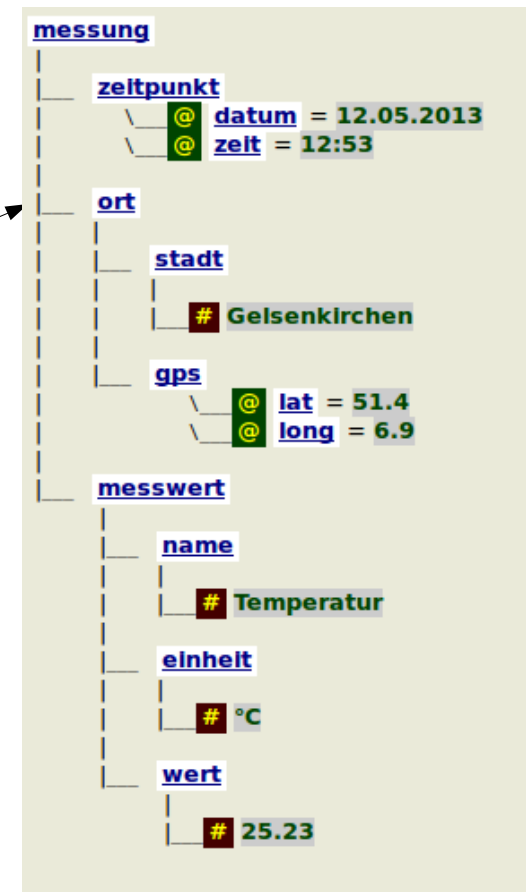
# XML → Baum



- Elementknoten (Ast): ein Knoten, der weitere Knoten enthalten kann (Kindknoten, „child nodes“). Aus jedem Tag wird ein Elementknoten

```
<?xml version='1.0' encoding='UTF-8'?>
<messung>
  <zeitpunkt datum='12.05.2013' zeit='12:53' />
  <ort>
    <stadt>Gelsenkirchen</stadt>
    <gps lat='51.4' long='6.9' />
  </ort>
  <messwert>
    <name>Temperatur</name>
    <einheit>°C</einheit>
    <wert>25.23</wert>
  </messwert>
</messung>
```

Elementknoten



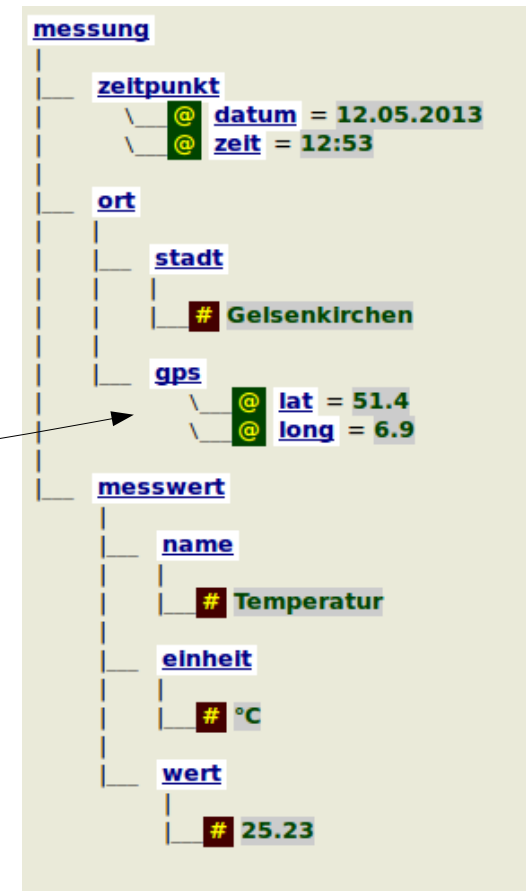
# XML → Baum



- Attributknoten (Blatt): jedes Attribut bildet einen eigenen Knoten. Der Wert des Attributs bildet *keinen* eigenen Knoten, sondern steckt als „value“ im Knoten.

```
<?xml version='1.0' encoding='UTF-8'?>
<messung>
  <zeitpunkt datum='12.05.2013' zeit='12:53' />
  <ort>
    <stadt>Gelsenkirchen</stadt>
    <gps lat='51.4' long='6.9' />
  </ort>
  <messwert>
    <name>Temperatur</name>
    <einheit>°C</einheit>
    <wert>25.23</wert>
  </messwert>
</messung>
```

Attributknoten



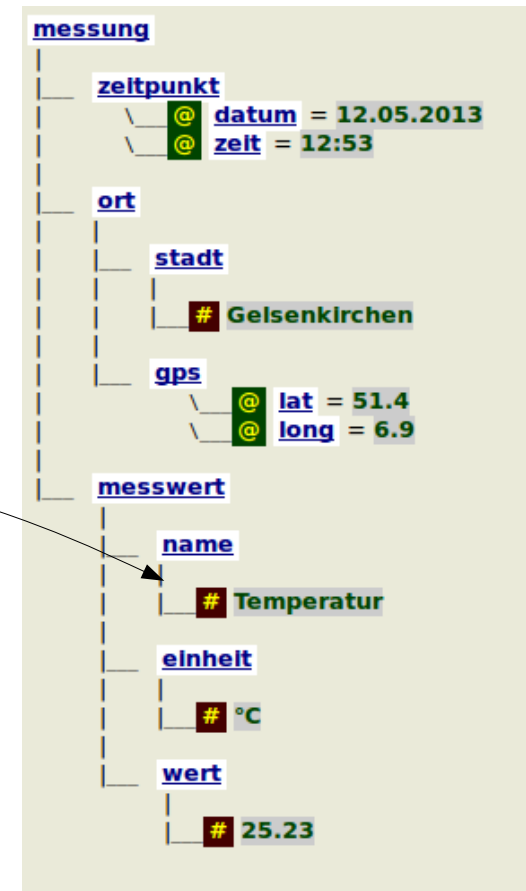
# XML → Baum



- Textknoten (Blatt): der Text eines Elements wird zu einem eigenen Textknoten.

```
<?xml version='1.0' encoding='UTF-8'?>
<messung>
  <zeitpunkt datum='12.05.2013' zeit='12:53' />
  <ort>
    <stadt>Gelsenkirchen</stadt>
    <gps lat='51.4' long='6.9' />
  </ort>
  <messwert>
    <name>Temperatur</name>
    <einheit>°C</einheit>
    <wert>25.23</wert>
  </messwert>
</messung>
```

Textknoten



# Knoten genau Unterscheiden

---



Bitte verinnerlichen und gut unterscheiden!

- Elemente, Elementknoten
  - ein Tag, also etwas, das selber wieder Kindknoten enthalten kann
- Attributknoten
  - aus jedem Attribut wird ein Attributknoten. Hat natürlich keine Kindknoten, ist aber immer Kind eines Elements
- Textknoten
  - aus jedem Text innerhalb eines Tags wird ein Textknoten. Hat natürlich keine Kindknoten, ist aber immer Kind eines Elements

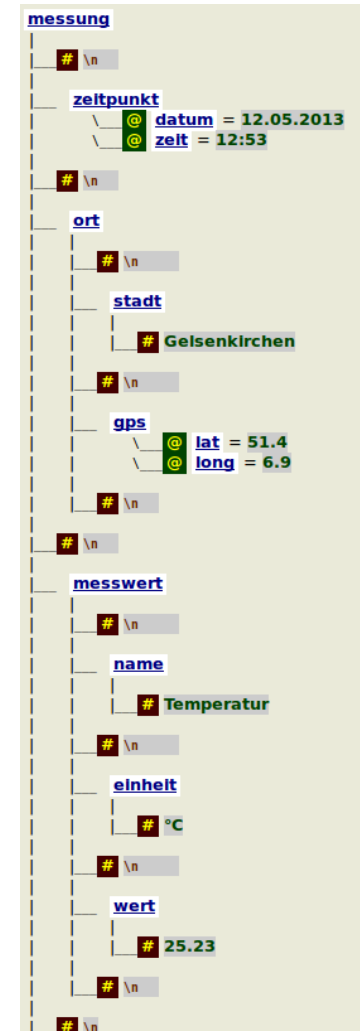


# Unerwartete Textknoten



- Watt'n datt ?
- ich gebe es zu – auf den vorherigen Folien habe ich etwas beschönigt....

```
<messung>
  <zeitpunkt datum='12.05.2013'
             zeit='12:53' />
  <ort>
    <stadt>Gelsenkirchen</stadt>
    <gps lat='51.4' long='6.9' />
  </ort>
  <messwert>
    <name>Temperatur</name>
    <einheit>°C</einheit>
    <wert>25.23</wert>
  </messwert>
</messung>
```

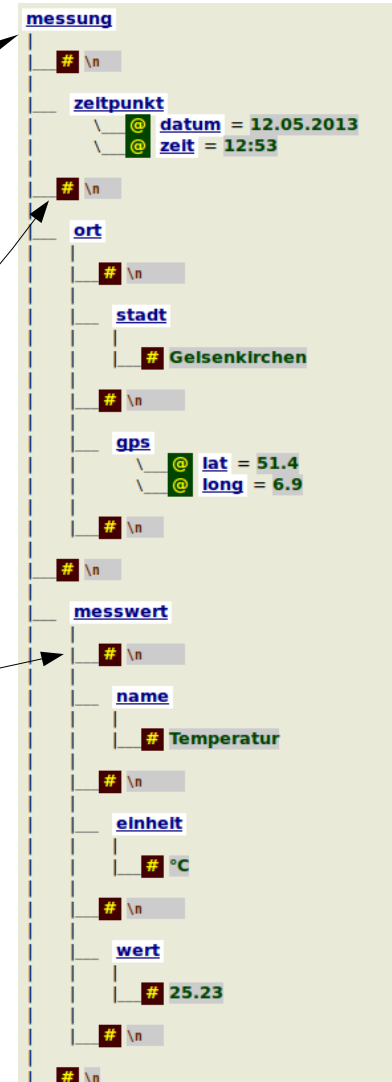


# Unerwartete Textknoten



- Zeilenumbrüche (\n) und Leerzeichen sind auch Text und bilden daher (ungewollte) Textknoten!

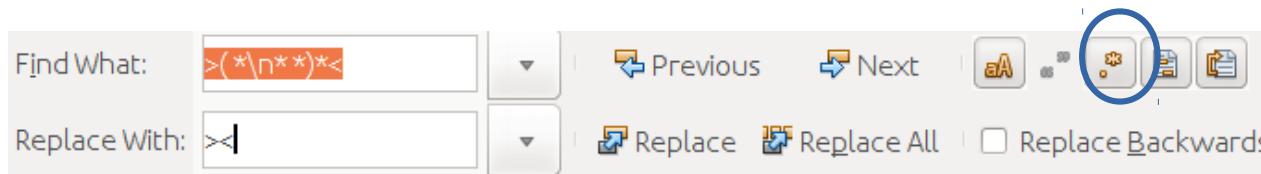
```
<messung>
  <zeitpunkt datum='12.05.2013'
             zeit='12:53' />
  <ort>
    <stadt>Gelsenkirchen</stadt>
    <gps lat='51.4' long='6.9' />
  </ort>
  <messwert>
    <name>Temperatur</name>
    <einheit>°C</einheit>
    <wert>25.23</wert>
  </messwert>
</messung>
```



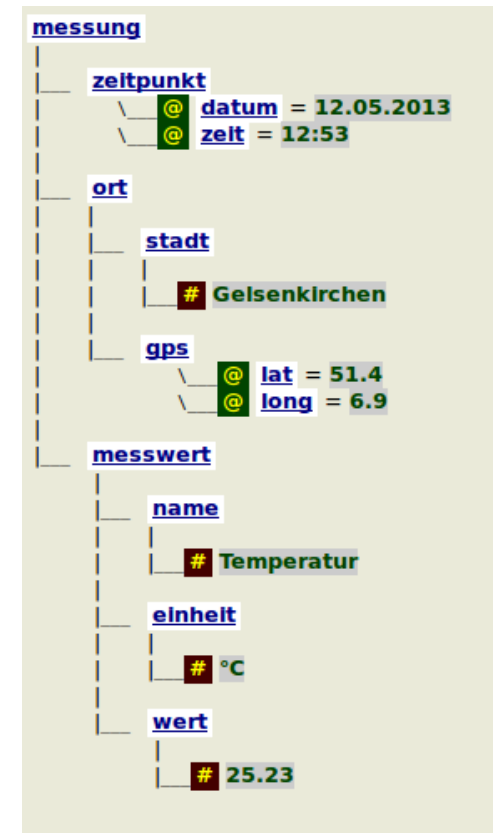
# Baum ohne „line-break-nodes“



- wer die zusätzlichen Knoten nicht will, muss im XML Dokument alle line-breaks und Leerzeichen (!) zwischen den Tags entfernen
- z.B. durch Suchen&Ersetzen mit einer Regex<sup>1)</sup>



```
<?xml version='1.0' encoding='UTF-8'?><messung><zeitpunkt datum='12.05.2013' zeit='12:53' /><ort><stadt>Gelsenkirchen</stadt><gps lat='51.4' long='6.9' /></ort><messwerte><messwert><name>Temperatur</name><einheit>°C</einheit><wert>25.23</wert></messwert></messwerte></messung>
```

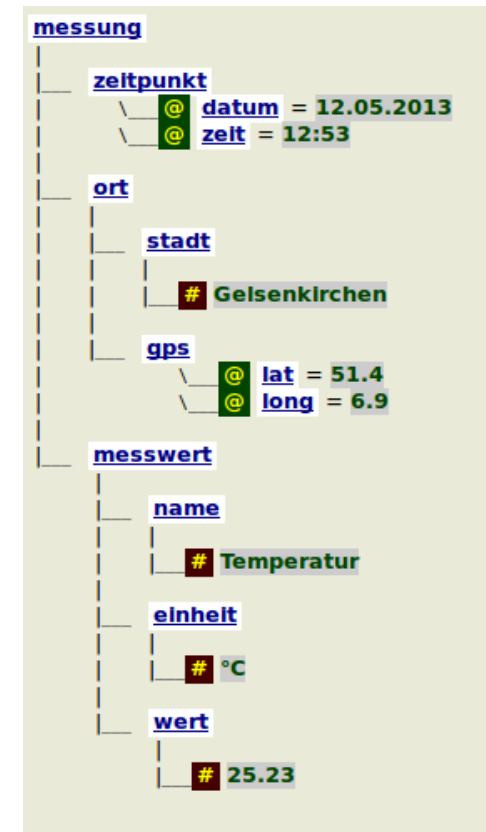


1) Achten Sie auf das Leerzeichen vor dem ersten und dritten Stern....

# Ein XPath



- Was bekommt man für  
`/messung/messwert/einheit/text()`
- Ihr Tipp:



# XPath - Einzelteile



- Mit einem XPath wählt man Knoten aus einem XML-Dokument aus. Ein XPath besteht aus einem oder mehreren „Lokalisierungsschritten“

**/messung/messwert/einheit/text()**

/ am Anfang:  
Pfad beginnt beim  
Wurzel-Element

**messung/messwert**  
wähle alle „messwert“  
Elemente, die Kindsnoten  
von „messung“ sind

**einheit/text()**  
wähle alle Textknoten, die  
Kinder von „einheit“ sind

# XPath = Lokalisierungspfad



- Mit einem XPath wählt man Knoten aus einem XML-Dokument aus.
- Ein XPath besteht aus einem oder mehreren „Lokalisierungsschritten“
- man erhält als Ergebnis entweder einen Teilbaum oder einen Wert
- die vorherige Darstellung war eine abgekürzte Fassung, die ungekürzte Form ist

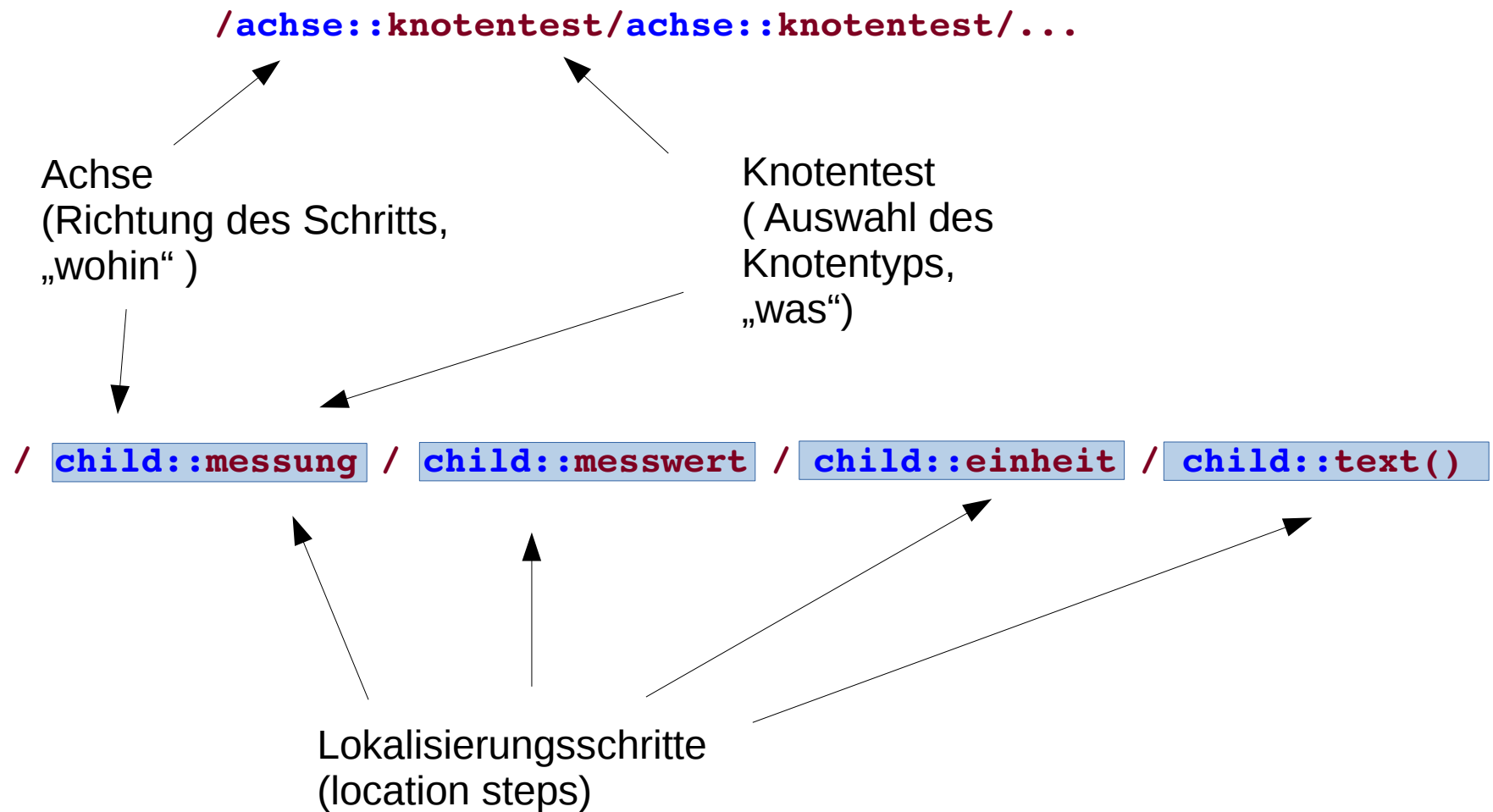
`/child::messung/child::messwert/child::einheit/child::text()`

Lokalisierungspfad

# Lokalisierungsschritte



- die Lokalisierungsschritte bestehen aus



# Knotentests



Syntax	wählt	ergibt
<elementname>	Elementknoten	alle „messwert“ Knoten, die Kind von „messung“ sind
*	alle Elementknoten	alle Elementknoten, die Kind von „messung“ sind
text()	Textknoten	die Textknoten, die Kind von „messung“ sind
node()	Knoten	alle Element- und Textknoten die Kind von „messung“ sind

Enter Path

/child::messung/child::messwert

Enter Path

/child::messung/child::\*

Your path selects 4 Elements

Enter Path

/child::messung/child::node()

Your path selects 9 Elements

/

messung

@titel=test

#text:a

zeitpunkt

@datum=12.05.2013

@zeit=12:53

#text:

ort

#text:

stadt

#text:

gps

#text:

#text:

messwert

#text:

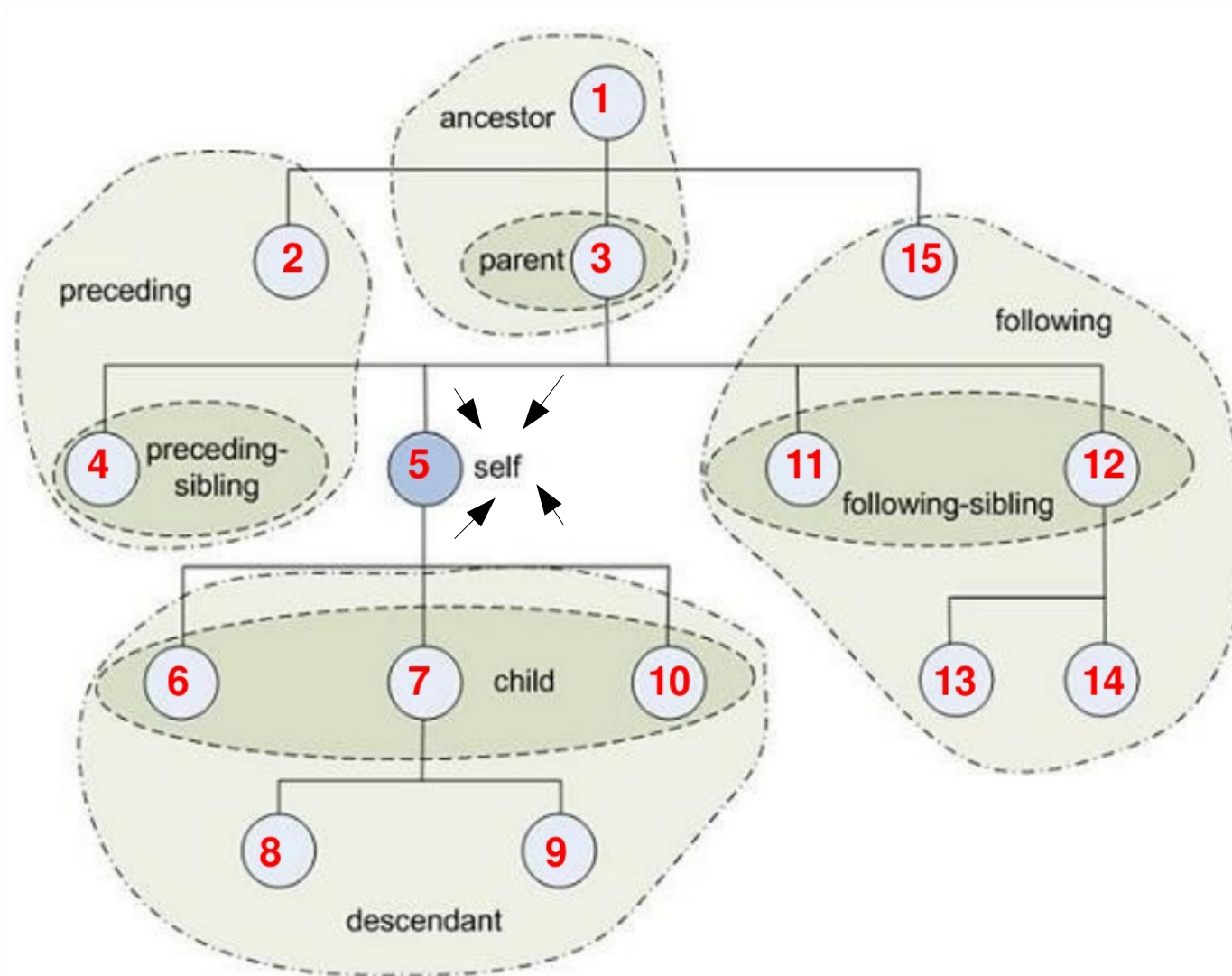


# Knotentests



Syntax	wählt	Beispiel	ergibt
<elementname>	Elementknoten	<code>/child::messung/child::messwert</code> <code>/messung/messwert</code>	alle „messwert“ Knoten, die Kind von „messung“ sind
*	alle Elementknoten	<code>/child::messung/child::*</code> <code>/messung/*</code>	alle Elementknoten, die Kind von „messung“ sind
text()	Textknoten	<code>/child::messung/child::text()</code> <code>/messung/text()</code>	die Textknoten, die Kind von „messung“ sind
node()	Knoten	<code>/child::messung/child::node()</code> <code>/messung/node()</code>	alle Element- und Textknoten die Kind von „messung“ sind

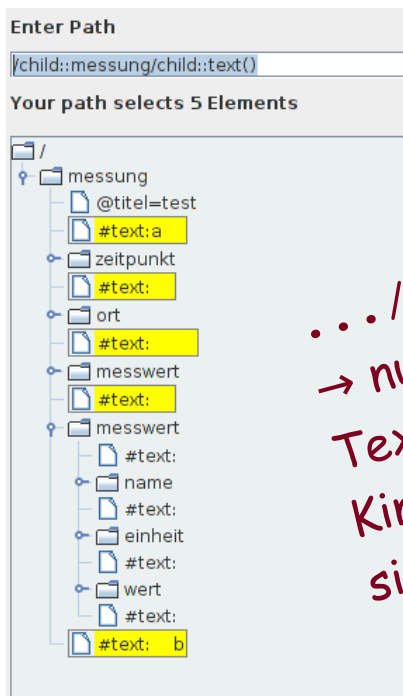
# Verwandtschaftsverhältnisse



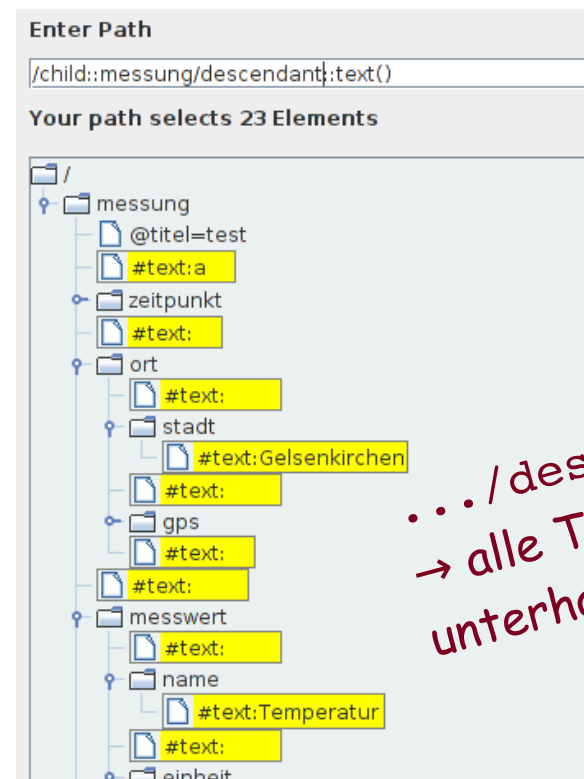
# Descendant Achsen



- Die Achsen `descendant::` und `descendant-or-self::` sind die zweitwichtigsten Achsen nach `child::`
  - mit einem Schritt durchsucht man den kompletten Unterbaum eines Knotens



*.../child::text()  
→ nur die  
Textknoten, die  
Kinder von „messung“  
sind*



*.../descendant::text()  
→ alle Textknoten  
unterhalb von „messung“*

# Descendant abgekürzt



- die Achse descendant:: darf mit / abgekürzt werden

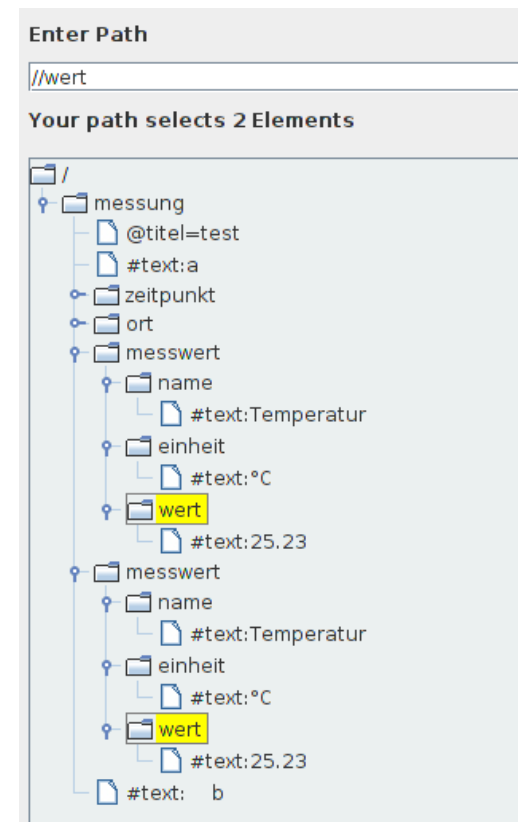
`/child::messung/descendant::node()`

ist abgekürzt `/messung//node()`

- man kann mit den descendant-Achsen auch das ganze Dokument auf einmal durchsuchen

`/descendant::wert`

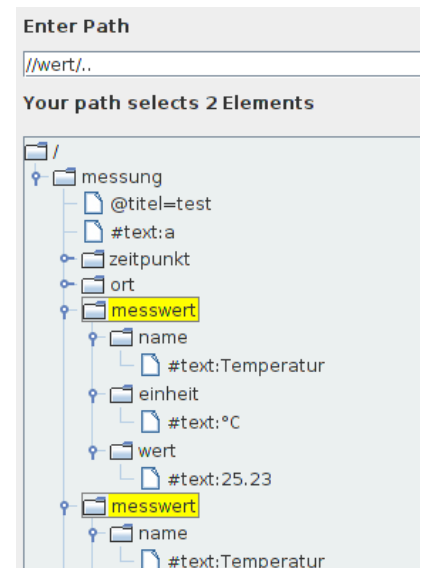
`//wert`



# Die Parent-Achse



- ein Schritt nach „oben“
- Ergebnis ist ein Elementknoten (nur Elemente haben Kinder)
- `/descendant::text()/parent::einheit`  
→ sucht erst alle Textknoten und dann davon die Elternknoten, die vom Typ „einheit“ sind
- oft braucht man einfach nur alle Elternknoten
- `/descendant::wert/parent::node()`
- `parent::node()` wird mit `..` abgekürzt  
`//wert/..`
- Achtung: hier wird Achse und Knotentest mit einer Abkürzung zusammengefasst !



# Attribute



- Attribute sind komisch: sie sind Knoten, weil sie einen „parent“ haben, aber sie sind kein Knoten, weil sie weder über `child::*` noch über `child::node()` ausgewählt werden können.
- stattdessen haben Attribute ihre eigene Achse:  
`/child::messung/attribute::titel`
- die Attributachse kann mit `@` abgekürzt werden  
`/messung/@titel`
- als Knotentest für die Attributachse gibt es nur `<attributname>` und `*` (=alle)

# XPath - Werkzeuge

---

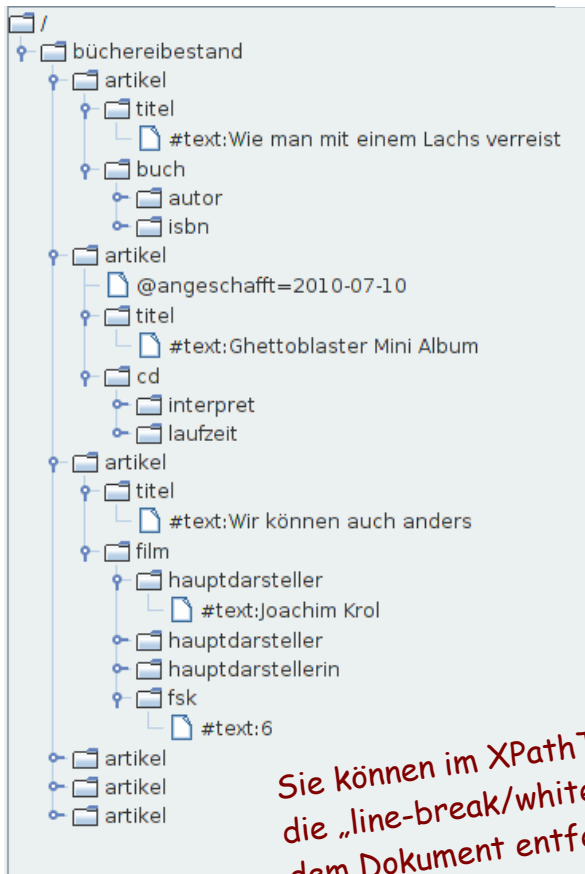


- JXPathTester  
<http://web.wara.de/~mertens/fortbildungxml/tools> oder T:\..\n  
(Starten mit Doppelklick oder  
    `java -jar JXPathTester.jar`)
- <http://xmlgrid.net/xpath.html> (online testen)
- Firefox-AddOn: XPath Checker
- Netbeans XPath-Plugin

# Datei für die Aufgaben



- Öffnen Sie im JXPath-Tester die Datei `buechereibestand_xpath_uebung.xml`



```
<buechereibestand>
  <artikel>
    <titel>Wie man mit einem Lachs verreist</titel>
    <buch>
      <autor>Umberto Eco</autor>
      <isbn>1231245-231</isbn>
    </buch>
  </artikel>
  <artikel angeschafft="2010-07-10">
    <titel>Ghettoblaster Mini Album</titel>
    <cd>
      <interpret>Street Sweeper Social Club</interpret>
      <laufzeit>00:35:45</laufzeit>
    </cd>
  </artikel>
  <artikel>
    <titel>Wir können auch anders</titel>
    <film>
      <hauptdarsteller>Joachim Krol</hauptdarsteller>
      <hauptdarsteller>Moritz Kipp</hauptdarsteller>
      <hauptdarstellerin>Sophie Rois</hauptdarstellerin>
      <fsk>6</fsk>
    </film>
  </artikel>
  <!-- ... -->
</buechereibestand>
```

Sie können im XPathTester einstellen, dass die „line-break/whitespace-Textknoten“ aus dem Dokument entfernt werden.



# Übung zu Achsen und Knotentests

---



- wählen Sie alle Filmknoten aus
- geben Sie die Interpreten (die konkreten Interpreten, nicht die Knoten) aller verfügbaren CDs aus
- zeigen Sie die Titel aller Artikel an
- finden Sie die Titel aller Bücher
- suchen Sie alle Artikel, die eine Angabe zur Laufzeit haben und lassen Sie deren Titel anzeigen
- geben sie die Titel aller Artikel an, zu denen es das „angeschafft“ Attribut gibt
- finden Sie Filmknoten, für die Hauptdarstellerinnen angegeben sind

# Übung zu Achsen und Knotentests



- wählen Sie alle Filmknoten aus  
`/artikel/film`
- geben Sie die Interpreten (die konkreten Interpreten, nicht die Knoten) aller verfügbaren CDs aus  
`/büchereibestand/artikel/cd/interpret/text()`
- zeigen Sie die Titel aller Artikel an  
`/büchereibestand/artikel/titel/text()`
- finden Sie die Titel aller Bücher  
`//buch/../titel/text()`
- suchen Sie alle Artikel, die eine Angabe zur Laufzeit haben und lassen Sie deren Titel anzeigen  
`//laufzeit/../..titel/text()`
- geben sie die Titel aller Artikel an, zu denen es das „angeschafft“ Attribut gibt  
`//@angeschafft/../titel/text()`
- finden Sie Filmknoten, für die Hauptdarstellerinnen angegeben sind  
`//hauptdarstellerin/..`



# XPath - Prädikate

- jeder Lokalisierungsschritt kann optional durch ein Prädikat eingegrenzt werden (= Filter)
- Prädikate werden in eckigen Klammern angegeben

```
.../achse::knotentest[prädikat]/achse::knotentest[prädikat]/...
```

- also z.B.

```
/descendant::ort/child::gps[attribute::lat>46]/...
```

```
//ort/gps[@lat>46]/...
```

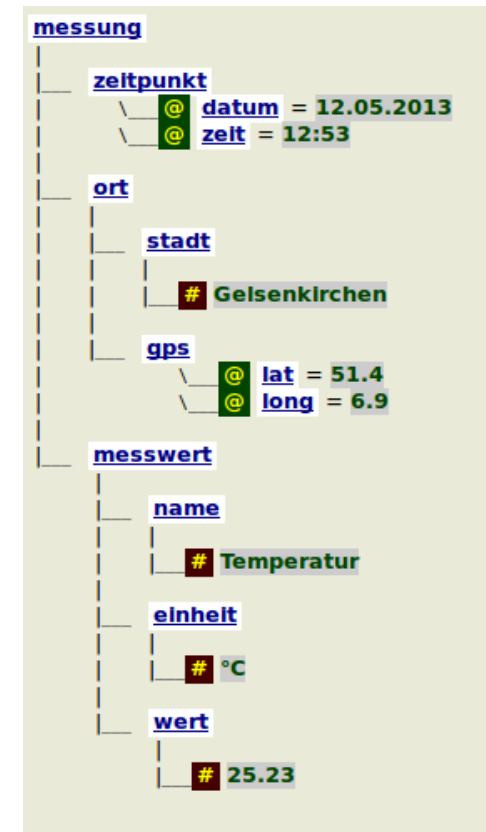
→ „wähle alle nur „gps“ Knoten, bei denen das Attribut „lat“ größer ist als 46“

- das Prädikat kann Knotenwerte (Attributswerte oder Textknoteninhalt) vergleichen und/oder Prädikatsfunktionen anwenden
- Liste aller Funktionen (lang...) :  
<http://wiki.selfhtml.org/wiki/XML/XSL/XPath/Funktionen>

# Beispiele für Prädiakte



- `//messwert[1]` → erstes „messwert“ Element  
(kurz für `//messwert[position() = 1]` )
- `/messung/messwert[last()]` → letztes „messwert“ Element
- `///messwert[last()-1]` → vorletztes „messwert“ Element
- `/messung/messwert[position()>1]` → alle „messwert“ Elemente außer dem ersten
- `/messwert/ort[stadt]` → alle „ort“ Elemente, die ein „stadt“ Element enthalten
- `//einheit[@praefix]` → alle „einheit“ Elemente, die ein Attribut „praefix“ enthalten
- `//gps[@lat>46]` → alle „gps“ Elemente, deren Attribut „lat“ größer als 46 ist



# Komplexere Pfade

---



```
//gps[@lat>46] / ../stadt/text()
```

```
/descendant::gps[attribute::lat>46]/parent::node()/child::stadt/child::text()
```

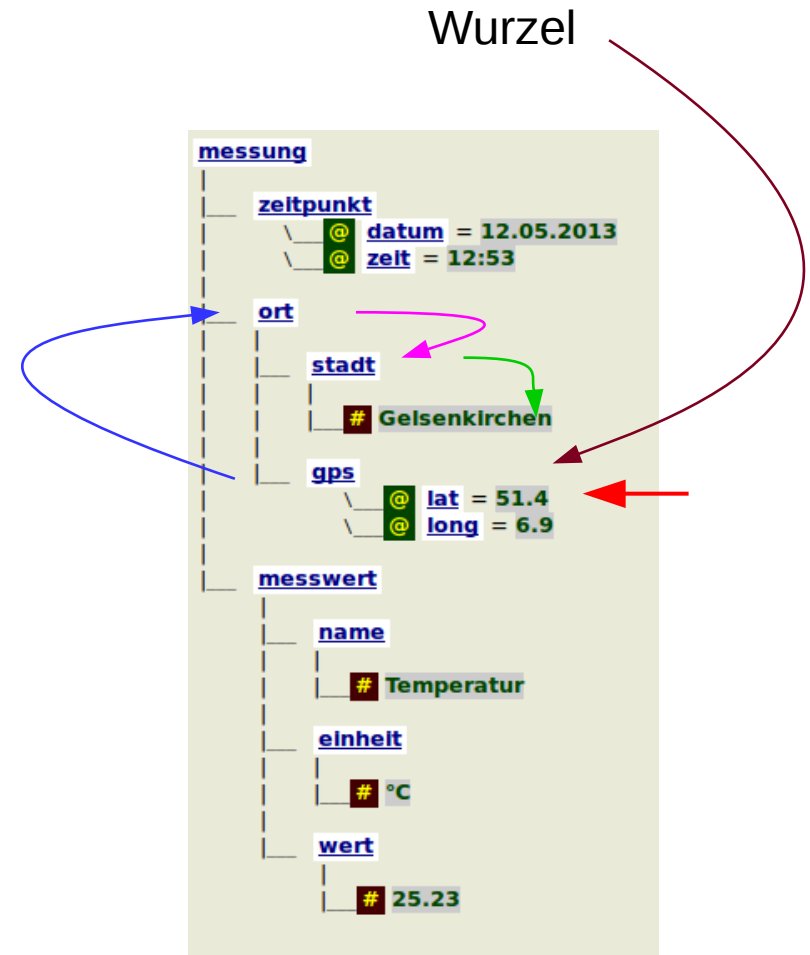
# Komplexer XPath - Auflösung



```
/descendant::gps[attribute::lat>46]/parent::node()/child::stadt/child::text()
```

```
//gps[@lat>46]/../stadt/text()
```

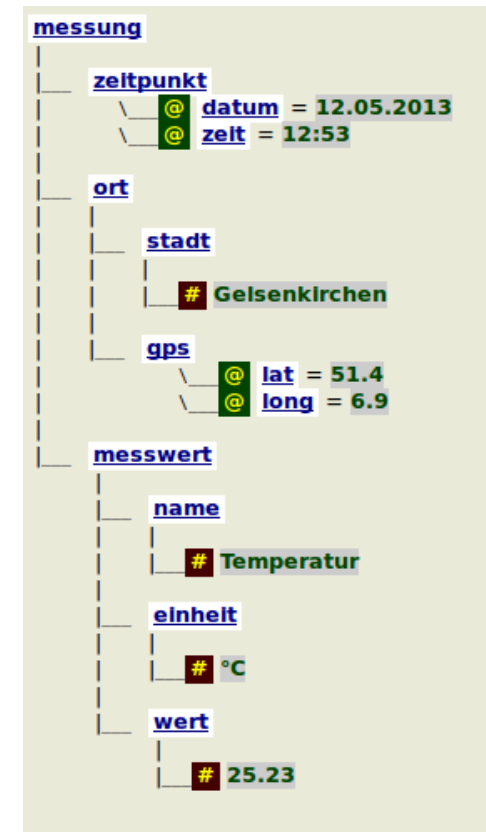
- „suche von der Wurzel aus beliebig tief nach den gps-Knoten,  
bei denen das Attribut „lat“ größer ist als 46,  
gehe zu den Elternknoten,  
suche deren „Stadt“-Kinds-knoten,  
und gib mir deren Text“



# Beispiele für Vergleich von Textknoteninhalten



- `//messwert[wert>20 and einheit != 'bar']`
  - alle „messwert“ Elemente,  
die ein Kindelement „wert“ enthalten,  
dessen Inhalt > 20 ist
  - und
  - deren Kindelement „einheit“  
nicht den Text „bar“ enthält
- gibt man im Prädikat einen Elementnamen an, so bezieht man sich auf ein Kindelement des aktuellen Elements
- verglichen wird der Inhalt des Textknotens dieses Kindelements
- gibt es das Kindelement nicht oder enthält das Kindelement keinen Textknoten so ist der Vergleich „false“
- lässt sich der Inhalt des Textknotens nicht in den Datentyp umwandeln, der für den Vergleich notwendig ist, ist der Vergleich ebenfalls „false“



# Prädikate-Aufgaben (mit buechereibestand)

---



- geben Sie den Titel des letzten Artikels im Bestand aus
- zeigen Sie Artikel an, die im letzten Jahrtausend angeschafft wurden
- finden Sie Filme, für die mehr als ein Hauptdarsteller angegeben ist
- lassen Sie anzeigen, wann das „Ghettoblaster Mini Album“ angeschafft wurde
- finden Sie Filme, für die es einen Hauptdarsteller und eine Hauptdarstellerin gibt und geben Sie deren Altersfreigabe an
- formulieren Sie für Ihren Nachbarn fiesere Aufgaben, als ich das bisher getan habe....
- auf der Seite <http://learn.onion.net/language=de/9075/w3c-xpath> finden Sie weitere, sehr schön aufbereitete Übungen (Sie werden sogar für richtige Antworten gelobt... )
- suchen Sie die CD-Knoten der CDs, die kürzer als 50min sind



# Prädikate-Aufgaben (mit buechereibestand)



- geben Sie den Titel des letzten Artikels im Bestand aus  
`//artikel[last()]`
- zeigen Sie Artikel an, die im letzten Jahrtausend angeschafft wurden  
`//artikel[@angeschafft and number(substring(@angeschafft,1,4))<'2000']`
- finden Sie Filme, für die mehr als ein Hauptdarsteller angegeben ist  
`//film[(count(hauptdarsteller)+count(hauptdarstellerin))>1]`
- lassen Sie anzeigen, wann das „Ghettoblaster Mini Album“ angeschafft wurde  
`//artikel[titel='Ghettoblaster Mini Album']/@angeschafft`
- finden Sie Filme, für die es einen Hauptdarsteller und eine Hauptdarstellerin gibt und geben Sie deren Altersfreigabe an  
`//film[hauptdarsteller and hauptdarstellerin]/fsk`
- formulieren Sie für Ihren Nachbarn fiesere Aufgaben, als ich das bisher getan habe....
- auf der Seite <http://learn.onion.net/language=de/9075/w3c-xpath> finden Sie weitere, sehr schön aufbereitete Übungen (Sie werden sogar für richtige Antworten gelobt... )
- suchen Sie die CD-Knoten der CDs, die kürzer als 50min sind  
`//cd[number(substring(laufzeit,4,2))<50]`

# Mehr Material

---



- <http://parscube.de/xml-technologien/xpath/>
- [http://www.w3schools.com/xpath/xpath\\_syntax.asp](http://www.w3schools.com/xpath/xpath_syntax.asp)
- <http://www.teialehrbuch.de/Kostenlose-Kurse/XML-XSL/33835-Lokalisierungspfade.html>
- <http://wiki.selfhtml.org/wiki/XML/XSL/XPath/Funktionen>
- <http://www.informit.com/articles/article.aspx?p=102644>
- <http://learn.onion.net/language=de/9075/w3c-xpath>



**Danke für Ihre Aufmerksamkeit**

**Fragen ?**