



XML Information Set

<http://www.w3.org/TR/xml-infoset>



XML Information Set

Übersicht

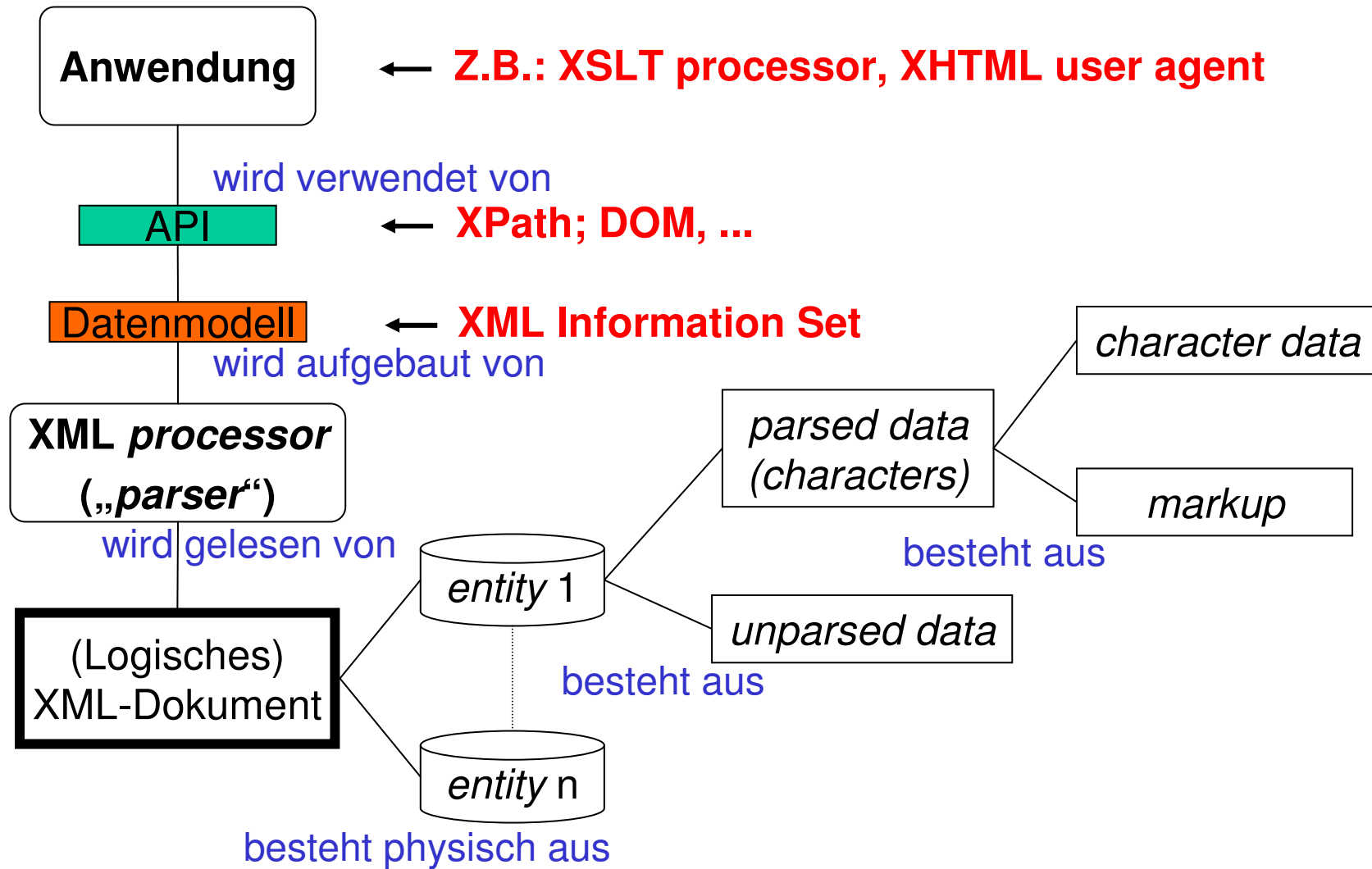
Begriffsbildung, Beispiel

Item Properties

Beispiel, Was nicht enthalten ist



Einordnung: XML Infoset, XPath





- **Was ist XML Information Set?**

- Die Spezifikation abstrakter Datenstrukturen, die den Anwendungen zugänglichen Inhalt von wohlgeformten (nicht unbedingt auch „gültigen“) XML-Dokumenten beschreiben.
- Eine formale Beschreibung des „Outputs“ von XML-Prozessoren
- Eine abstrakte Sicht, unabhängig von spezifischen APIs
- Wie bei W3C üblich: Eine Empfehlung

- **Was ist ein XML Infoset?**

- Meistens: Eine baumartige Repräsentation eines XML-Dokuments.
- Auch: Eine künstlich generierte Datenstruktur ohne zugrundeliegendes XML-Dokument, die den Spezifikationen von XML Information Set genügt.

- **Was ist XML Infoset nicht?**

- Erschöpfend (im Sinne aller Informationen eines XML-Dokuments)
- Eine Minimalanforderung an den Output von XML-Prozessoren.

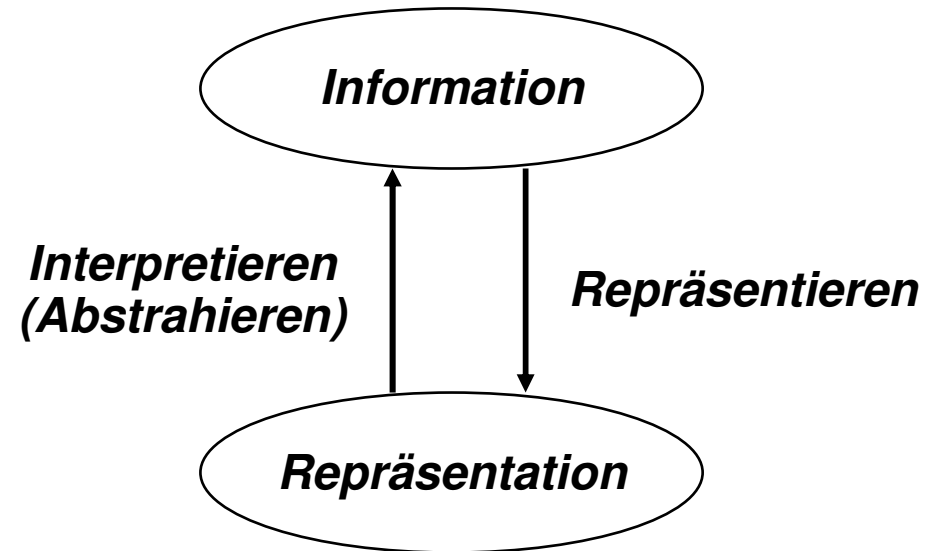


- **Entwicklung von XML Information Set**

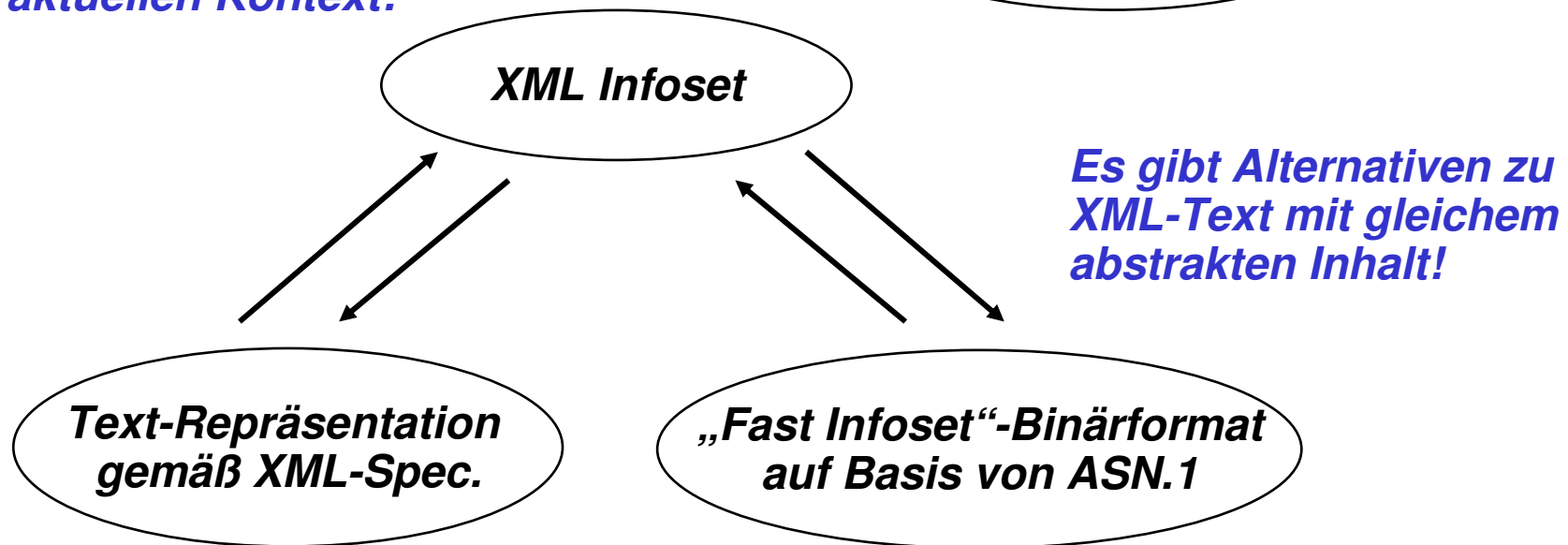
- Quelle: W3C
- Status: Empfehlung (REC)
- Dokumentation:
<http://www.w3.org/TR/xml-infoset>
- Stand:
 - 2001-10-24 1. Auflage
 - 2004-02-04 2. Auflage, berücksichtigt
XML 1.1 und Namespaces 1.1



Erinnerung - allgemein gilt:



Anwendung auf den aktuellen Kontext:





- **Begriffsbildung**

- ***Information set*** (Informationssatz)
 - Entspricht einem Datenbaum (*tree*)
 - Enthält genau ein *document item* und weitere *information items*.
- ***Information item*** (Ein Element des Infosatzes)
 - Entspricht einem Knoten (*node*) in diesem Baum
 - Enthält „Eigenschaften“ (*properties*), z.B. Listen weiterer *info items*.
- Die Begriffe *tree* und *node* wurden vermieden, um zu betonen, dass Zugriffe auf den *information set* auch über andere Schnittstellen als baumartige erfolgen können, z.B. ereignisgesteuerte (*event-based*) oder anfrageartige (*query-based*).

- **Vorsicht:**

- Elemente des Infosatzes lassen sich **nicht** 1:1 identifizieren mit Knoten des DOM oder Knoten bzw. Teilbäumen von XPath (obwohl dies auf weiten Strecken sehr wohl möglich ist).



- **Unterscheide folgende Werte von Eigenschaften voneinander:**
 - „*unbekannt*“ (*unknown*)
 - „*kein Wert*“ (*no value*)
 - Leere Zeichenkette / Liste / Menge
- **Bemerkungen**
 - Einem Attributwert kann die leere Zeichenkette zugewiesen werden. Dies ist ein anderer Inhalt als das Ausbleiben jeglicher Zuweisung!
 - In anderen Notationen werden die Schlüsselwörter „*null*“ oder „*nil*“ verwendet. Dies vermeidet die XML Infoset-Spezifikation, um Verwechslungen vorzubeugen, jedoch besteht konzeptionell eine enge Verwandtschaft.



- **Übersicht: Typen von *Information Items***
 - *Document*
 - *Element*
 - *Attribute*
 - *Unexpanded entity reference*
 - *Character*
 - *Comment*
 - *Processing instruction*
 - *Notation*
 - *Unparsed entity reference*
 - *Namespace*



XML Infoset: Beispiel 1



- Ein einfaches XHTML 1.1-Beispiel

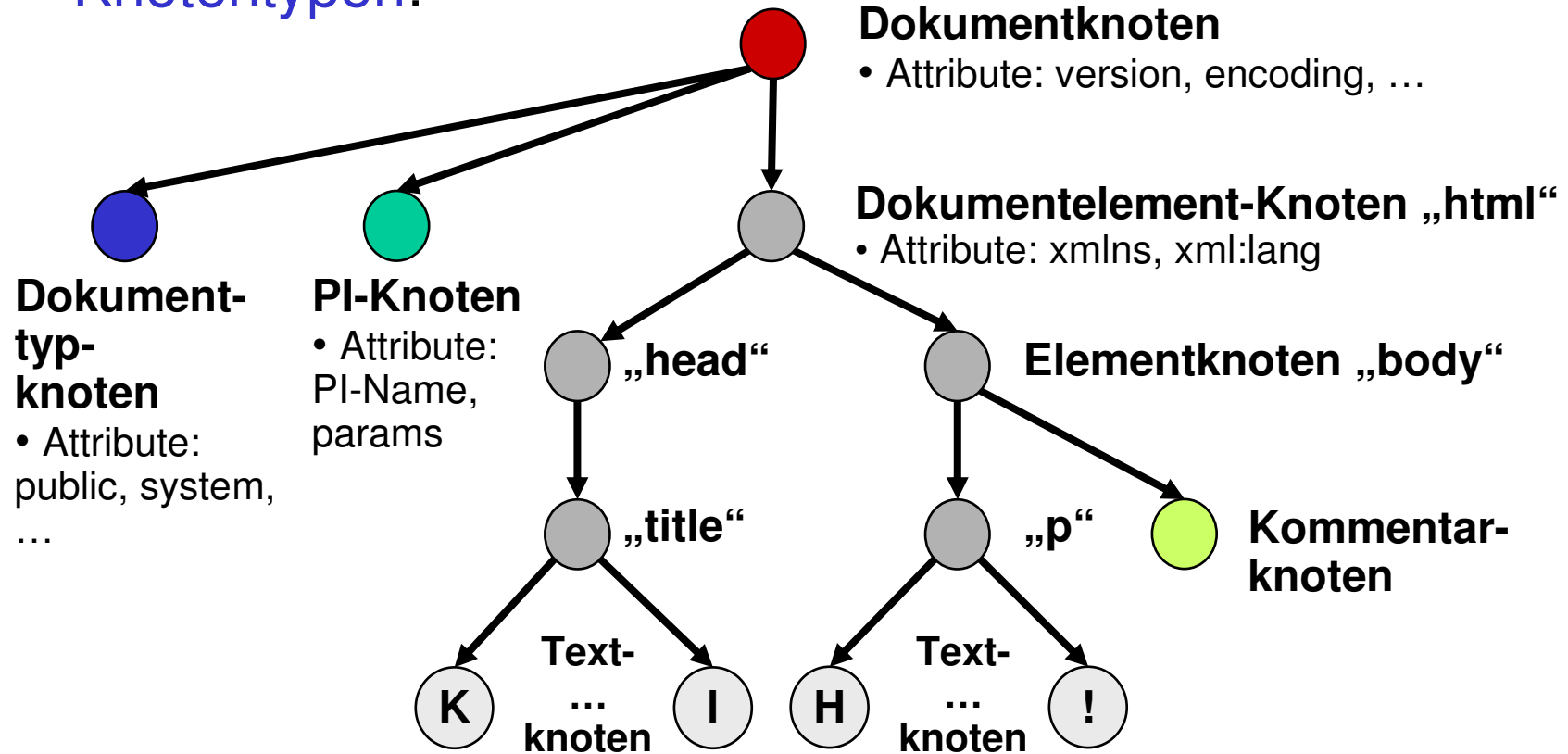
```
<?xml version="1.0" encoding="iso-8859-1" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
  "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<?xml-stylesheet href="hello.css" type="text/css"?>
<html
  xmlns="http://www.w3.org/1999/xhtml" xml:lang="de">
  <head>
    <title>Kleines XHTML-Beispiel</title>
  </head>
  <body>
    <p>Hallo, Welt!</p>
    <!-- Kommentar: Hier ergänzen! -->
  </body>
</html>
```



Ein Ordnungsansatz mit Graphen



- XML-Dokumente:
 - Markierte (attributierte), baumartige Graphen
 - Besitzen verschiedene Knotentypen:





XML Infoset: Beispiel 2



- **Beispiel:**

Anschreiben!

```
<?xml version="1.0"?>
<msg:message doc:date="19990421"
  xmlns:doc="http://doc.example.org/namespaces/doc"
  xmlns:msg="http://message.example.org/">
  Phone home!
</msg:message>
```

erzeugt folgendes XML Infoset:

- Ein document *information item*
- Ein element *info item*
- Ein attribute *info item*
- Drei namespace *info items*
- Zwei attribute *info items*, für die 2 *namespace-Attribute*
- Elf character *info items*, für Zeichenkette „Phone home!“



- **Document Information Item**

- ***children***

- Eine geordnete Liste, in Dokumentenreihenfolge

- Optional ein *document type declaration info item*

- Genau ein *element info item*

- Ein *PI info item* pro Vorkommen außerhalb des Dokumentenelements (Vorkommen innerhalb der DTD sind ausgenommen).

- ***document element***

- Das *element info item*, entspricht dem Dok.element.

- ***notations***

- Eine (ungeordnete) Menge von *notation info items*, je eins pro NOTATION-Deklaration in der DTD.



- **Document Information Item (Forts.)**

- ***unparsed entities***

Eine (ungeordnete) Menge von *unparsed entity info items*, je eins pro nicht vom Parser aufgelöster ENTITY-Deklaration in der DTD.

- ***base URI***

Base URI der *document entity*.

- ***character encoding scheme***

Der Name des *encoding*-Schemas, das dem *document entity* zugrunde liegt. Stammt aus der XML-Deklaration

- ***standalone***

Der Wert der *standalone*-Deklaration aus der XML-Deklaration, entweder „yes“ oder „no“. Optional, „kein Wert“ wenn fehlt.



- **Document Information Item (Forts.)**

- ***version***

Der Wert der *version*-Anweisung aus der XML-Deklaration.
Optional, „kein Wert“ wenn fehlt.

- ***all declarations processed***

Ein boolescher Wert, der nicht aus dem XML-Dokument selbst stammt, sondern anzeigt, ob der XML-Prozessor alle Deklarationen verarbeitet hat. Zwingend ist dies nur validierenden Parsern vorgeschrieben.



- Document info item **D1**:
 - children = (**E1**)
 - document element = **E1**
 - notations = (kein Wert)
 - unparsed entities = (kein Wert)
 - base URI = „file://...“ // irgendeine passende Quelle
 - character encoding scheme = „UTF-8“ // oder (kein Wert) ?
 - standalone = (kein Wert)
 - version = „1.0“
 - all declarations processed = true



- **Element Information Item**

- ***namespace name***

Der Name des Namensraums dieses Elements. Optional.

- ***local name***

Der Name des Elementtyps, ohne *namespace*-Präfix und Doppelpunkt.

- ***prefix***

Das *namespace*-Präfix des Elementtyp-Namens. Optional.

Bem.: Anwendungen, die *namespaces* unterstützen, sollten die Namensraum-Namen statt der Präfixwerte verwenden.



- **Element Information Item (Forts.)**

- ***children***

Eine (eventuell leere) geordnete Liste, in Dokumentenreihenfolge

Enthaltene *Info items* sind vom Typ:

element

processing instruction

unexpanded entity reference

character

comment

- ***attributes***

Eine (ungeordnete, ggf. leere) Menge von *attribute info items*, die entweder aus dem Element oder per *default*-Deklaration aus der DTD stammen.



- **Element Information Item (Forts.)**

- ***namespace attributes***

Eine (ungeordnete, ggf. leere) Menge von *attribute info items*, je eins pro *namespace*-Deklaration aus dem Element oder per *default*-Deklaration aus der DTD.

Per Definition lautet das *namespace* URI aller *namespace*-Attribute „http://www.w3.org/2000/xmlns“, auch die ohne Präfix („xmlns=...“)

- **base URI**

Base URI des Elements.

- **parent**

Der *document* bzw. *element info item*, dessen *children*-Liste den vorliegenden Eintrag enthält.



- **Element Information Item (Forts.)**

- ***in-scope namespaces***

Eine (ungeordnete) Menge von *namespace info items*, je eins für jeden Namensraum, der auf dieses Element wirkt.

Sie enthält immer einen Eintrag mit Präfix „`xml`“, der dem Namensraum „`http://www.w3.org/1998/namespace`“ implizit zugeordnet ist (→ Vorlesungseinheit zu Namensräumen).

Sie enthält nie einen Eintrag mit Präfix „`xmlns`“, denn eine Anwendung kann nie auf ein Element oder Attribut mit diesem Präfix stoßen.

Sie enthält je einen Eintrag, der dem aus der Liste zu „*namespace attributes*“ stammt - mit folgender Ausnahme: Einträge zu Deklarationen der Form „`xmlns: ' '`“, welchen den *default*-Namensraum „ent-deklarieren“.

Bemerkung: Zur Auflösung von Präfixwerten sollte diese Menge Vorrang haben vor der Verwendung von „*namespace attributes*“, da letztere Probleme erzeugen können bei künstlich generierten Infosets.



- Element info item **E1**:
 - namespace name = „http://message.example.org/“
 - local name = „message“
 - prefix = „msg“
 - children = (**C1**, **C2**, ..., **C11**)
 - attributes = { **A1** }
 - namespace attributes = { **AN1**, **AN2** }
 - in-scope namespaces = { **N3**, **N1**, **N2** }
 - base URI = „file://...“ // irgendeine passende Quelle
 - parent = **D1**



- **Attribute Information Item**

- ***namespace name***

Der Name des Namensraums dieses Attributs, sofern vorhanden.
Sonst „ohne Wert“.

- ***local name***

Der Name des Attributtyps, ohne *namespace*-Präfix und Doppelpunkt.

- ***prefix***

Das *namespace*-Präfix des Elementtyp-Namens, sonst „ohne Wert“.

Bem.: Anwendungen, die *namespaces* unterstützen, sollten die Namensraum-Namen statt der Präfixwerte verwenden.

- ***normalized value***

Der gemäß XML 1.0 normierte Attributwert.



- **Attribute Information Item (Forts.)**

- ***specified***

- Ein *flag*, das anzeigt, ob das Attribut tatsächlich im Element spezifiziert wurde (und nicht per default-Deklaration der DTD gefüllt wurde).

- ***attribute type***

- Gültige Werte sind ID, IDREF, ENTITY, ENTITIES, NMTOKEN, NMTOKENS, NOTATION, CDATA und ENUMERATION.

- „ohne Wert“, falls das Attribut nicht deklariert wurde,

- „unbekannt“, falls es eine ungelesene Deklaration gab.



- **Attribute Information Item (Forts.)**

- ***references***

- „ohne Wert“, falls ID, NMTOKEN, NMTOKENS, CDATA oder ENUMERATION.

- „unbekannt“, falls der Attributtyp „unbekannt“ ist
sonst (falls IDREF, IDREFS, ENTITY, ENTITIES, NOTATION):
Eine geordnete Liste der *info items* vom Typ
element, *unparsed entity* oder *notation*,
in der Reihenfolge ihres Erscheinens im Attributwert, bzw. „ohne Wert“ oder „unbekannt“ im Fall von Inkonsistenzen.

- ***owner element***

- Der *element info item*, aus dem das Attribut stammt.



- Attribute info item **A1**:
 - namespace name =
„http://doc.example.org/namespaces/doc“
 - local name = „date“
 - prefix = „doc“
 - normalized value = „19990421“
 - specified = „yes“ // „1“, true, ... ?
 - attribute type = (kein Wert)
 - references = (kein Wert) // ?
 - owner element = **E1**



- Attribute info item **AN1**:
 - namespace name = „http://www.w3.org/2000/xmlns“
 - local name = „doc“
 - prefix = „xmlns“ // ?
 - normalized value = „http://doc.example.org/namespaces/doc“
 - specified = „yes“ // „1“, true, ... ?
 - attribute type = (kein Wert)
 - references = (kein Wert) // ?
 - owner element = **E1** // ?



- Attribute info item **AN2**:
 - namespace name = „http://www.w3.org/2000/xmlns“
 - local name = „msg“
 - prefix = „xmlns“ // ?
 - normalized value = „ http://message.example.org/“
 - specified = „yes“ // „1“, true, ... ?
 - attribute type = (kein Wert)
 - references = (kein Wert) // ?
 - owner element = **E1** // ?



- **Processing Instruction Information Item**

- ***target***

Eine Zeichenkette, die den „*target*“-Teil der PI enthält (ein XML *name*).

- ***content***

Eine Zeichenkette, die den Inhaltsteil der PI enthält, ohne den *target*-Teil und führende *whitespace*-Zeichen. Ggf. die leere Zeichenkette.

- ***base URI***

Die base URI der PI.

- ***notation***

Falls verwendet: Der *notation info item* aus der PI. Sonst: „kein Wert“ bzw. „unbekannt“, je nachdem ob eine Deklaration fehlt oder eventuell nicht gelesen werden konnte.

- ***parent***

Der *info item*, in dessen *children*-Liste dieser *PI info item* erscheint.
Mögliche Typen sind: *document*, *element*, *document type definition*.



- **Unexpanded Entity Reference Information Item**

- ***name***

Eine Zeichenkette, die den Namen des *entity* enthält (ein XML *name*).

- ***system identifier***

Eine Zeichenkette mit dem angegebenen URI, ohne zusätzliche *escape*-Zeichen. Je nach Sachlage auch „ohne Wert“ oder „unbekannt“.

- ***public identifier***

Eine Zeichenkette mit dem angegebenen *public identifier* in normierter Darstellung. Je nach Sachlage auch „ohne Wert“ oder „unbekannt“.

- ***declaration base URI***

Base URI des *entity*, in dem die nicht aufgelöste *entity*-Referenz erscheint.

- ***parent***

Der *element info item*, in dessen *children*-Liste dieser *info item* erscheint.

Bemerkung:

Validierende XML-Prozessoren erzeugen keine solchen *info items*.



- **Character Information Item**

- ***character code***

- Der ISO 10646 / Unicode-Wert des Zeichens.

- ***element content whitespace***

- Eine boolesche Variable, „*true*“ falls das dargestellte Zeichen vom Typ *whitespace* ist, „*false*“ wenn nicht.

- Falls die Deklaration des zugrundeliegenden Elements nicht existiert bzw. nicht gelesen wurde, „ohne Wert“ bzw. „unbekannt“.

- Validierende XML-Prozessoren müssen diese Information stets liefern.

- ***parent***

- Der *element info item*, in dessen *children*-Liste dieser *info item* erscheint.

Bemerkungen

- Vorsicht - erheblicher *overhead* bei großen Dokumenten mit viel Freitext innerhalb von Elementen.
- Anwendungen steht es frei, die Zeichen wieder zu verketteten.
- Gerade hier gehen XPath und XSLT andere Wege!



- Character info item **C1**:
 - character code = „P“
 - element content whitespace = false
 - parent = **E1**
- ...
- Character info item **C6**:
 - character code = „“
 - element content whitespace = true
 - parent = **E1**
- ...
- Character info item **C11**:
 - character code = „!“
 - element content whitespace = false
 - parent = **E1**



- **Comment Information Item**

- ***content***

- Die Zeichenkette mit dem Kommentarinhalt.

- ***parent***

- Der *document* oder *element info item*, in dessen *children*-Liste dieser *info item* erscheint.

- **Bemerkungen**

- Kommentare innerhalb von DTDs gelangen nicht in den Infoset!



- **Document Type Declaration Information Item**

- ***system identifier***

Eine Zeichenkette mit dem in der DOCTYPE-Deklaration angegebenen URI des externen *Subsets*, ohne zusätzliche *escape*-Zeichen. Je nach Sachlage auch „ohne Wert“ oder „unbekannt“.

- ***public identifier***

Eine Zeichenkette mit dem angegebenen *public identifier* in normierter Darstellung. Je nach Sachlage auch „ohne Wert“ oder „unbekannt“.

- ***children***

Eine geordnete Liste der *PI info items* in der Reihenfolge, wie die entsprechenden PI in der DTD erscheinen.

PI im internen Subset erscheinen als erste.

- ***parent***

Der *document info item*.

Bemerkungen

- *entities* und *notations* werden vom *document info item* erfasst.



- **Unparsed Entity Reference Information Item**

- ***name***

Der Name des *entity*.

- ***system identifier***

Eine Zeichenkette mit dem in der Deklaration angegebenen URI.

- ***public identifier***

Eine Zeichenkette mit dem angegebenen *public identifier* in normierter Darstellung. Auch „ohne Wert“ oder „unbekannt“ mögl.

- ***declaration base URI***

Das base URI des *entity*, in dem die *entity*-Referenz erscheint.

- ***notation name***

Der Name der zugeordneten *notation*.

- ***notation***

Der *notation info item*, auf den die *entity*-Referenz verweist.



- **Notation Information Item**

- ***name***

- Der Name der *notation*.

- ***system identifier***

- Eine Zeichenkette mit dem in der NOTATION-Deklaration angegebenen URI, ggf. „ohne Wert“.

- ***public identifier***

- Eine Zeichenkette mit dem angegebenen *public identifier* in normierter Darstellung. Je nach Sachlage auch „ohne Wert“.

- ***declaration base URI***

- Das base URI des *entity*, in dem die NOTATION-Deklaration erscheint.



- **Namespace Information Item**

- ***prefix***

- Die mit einem *namespace* zu assoziierende Zeichenkette „ohne Wert“ im Fall des *default namespace* (Element-Präfix „xmlns:“).

- ***namespace name***

- Der zugeordnete Name des *namespace*.

- **Bemerkungen:**

- Fast alle *info items* entsprechen direkt bestimmten Objekten aus der XML 1.0-Spezifikation.

- Dieser hier nicht: Die *namespace*-Spezifikationen finden hier von Anfang an volle Unterstützung, nicht nur im Nachhinein.



- Namespace info item **N1**:
 - prefix = „msg“
 - namespace name = „http://message.example.org/“
- Namespace info item **N2**:
 - prefix = „doc“
 - namespace name =
„http://doc.example.org/namespaces/doc“
- Namespace info item **N3**:
 - prefix = „xml“
 - namespace name = „http://www.w3.org/1998/namespace“



XML Infoset

Das Eingangsbeispiel,
nun genauer



Das Eingangs-Beispiel 2 noch einmal:

```
<?xml version="1.0"?>
<msg:message doc:date="19990421"
  xmlns:doc="http://doc.example.org/namespaces/doc"
  xmlns:msg="http://message.example.org/">
  Phone home!
</msg:message>
```

Es erzeugt folgendes XML Infoset - diesmal genauer, nur ohne Verkettungen:

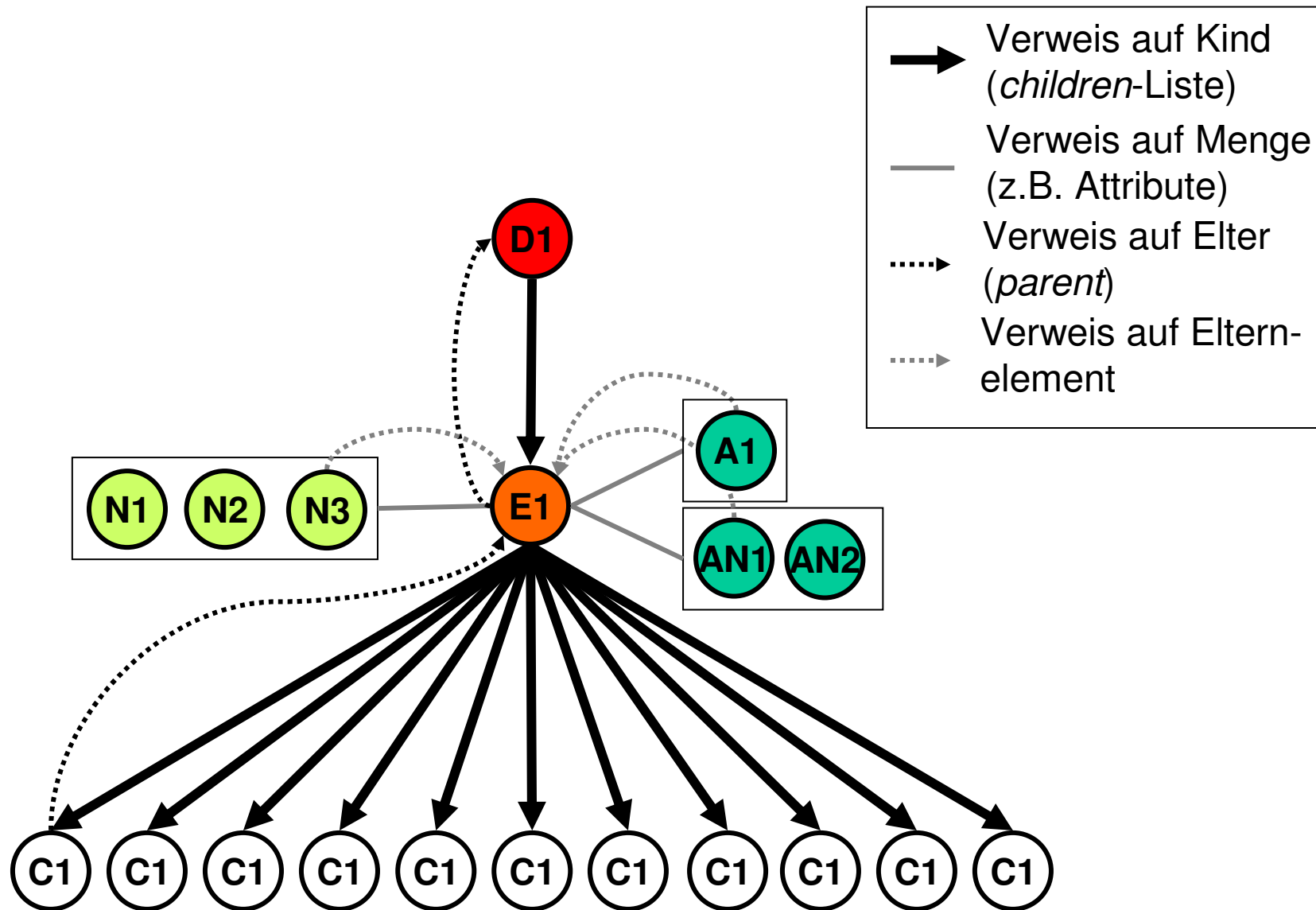
- Ein document *information item*
- Ein element *info item*
 - mit *namespace*-Eintrag „http://message.example.org/“
 - mit *local part* „message“ und Präfix „msg“
- (b.w.)



- Ein attribute *info item*
 - mit *namespace*-Eintrag
„http://doc.example.org/namespaces/doc“
 - mit *local part* „date“ und Präfix „doc“
 - mit dem normierten Wert „19990421“
- Drei namespace *info items*
 - für die Namensräume
„http://www.w3.org/XML/1998/namespace“,
„http://doc.example.org/namespaces/doc“ und
„http://message.example.org/“
- Zwei attribute *info items*, für die beiden *namespace-Attribute*
- Elf character *info items*, für die Zeichenkette „Phone home!“



XML Infoset: Verkettungen





XML Infoset

Was ein XML Infoset nicht enthält
Also: Was eine Anwendung nie über
Ihre XML-Dateien erfährt...



- Was ein XML Infoset nicht enthält
 - Folgen der Expansionen der XML-Prozessoren
 - Die Repräsentation von Zeichen
(direkt, per *char ref*, per *entity ref*, per *CDATA section*)
 - Die Grenzen von INCLUDE/IGNORE-Abschnitten in der DTD
 - Die Grenzen von *parameter entities* in der DTD.
 - Überlesene Deklarationen, z.B. innerhalb von IGNORE-Abschnitten
 - Die Grenzen von *general parsed entities*.
 - Die Grenzen von *CDATA sections*.
 - Folgen der Normierung der XML-Prozessoren
 - Die Art der Zeilenende-Codierung.
 - Die Art der verwendeten Anführungszeichen.



– Sonstiges

Die *content models* der Elementtyp-Deklarationen aus der DTD.
Gruppierung/Anordnung von Attributen laut ATTLIST-Deklaration.
Der Name des Dokumenttypen.

White space:

Außerhalb des *document element*

Der dem *target name* einer PI unmittelbar folgende

Innerhalb von *tags*.

Die Darstellung eines leeren Elements (<foo/> vs. <foo></foo>)

Die Reihenfolge der Attribute in einem *start tag*.

Die Reihenfolge der Deklarationen in der DTD.

Kommentare in der DTD.

Die Position von Deklarationen, z.B. intern vs. extern vs. indirekt per
parameter entity.

Die *default*-Werte von Attributen wie in der DTD deklariert.