Kapitel WT:III

III. Dokumentsprachen

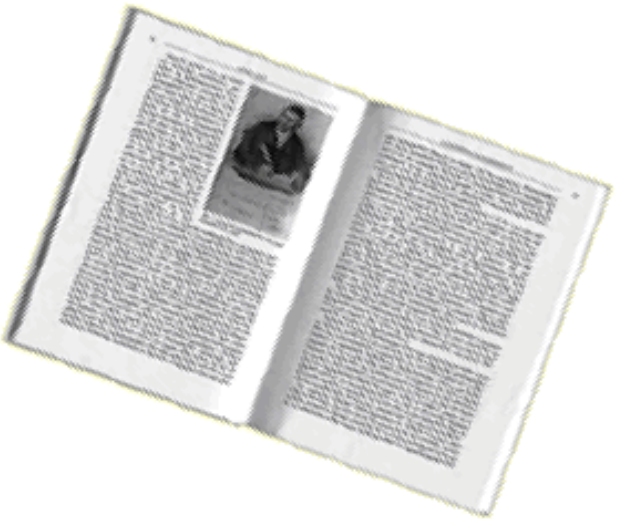
- Auszeichnungssprachen
- HTML
- Cascading Stylesheets CSS
- XML-Grundlagen
- XML-Schema
- Die XSL-Familie
- APIs für XML-Dokumente

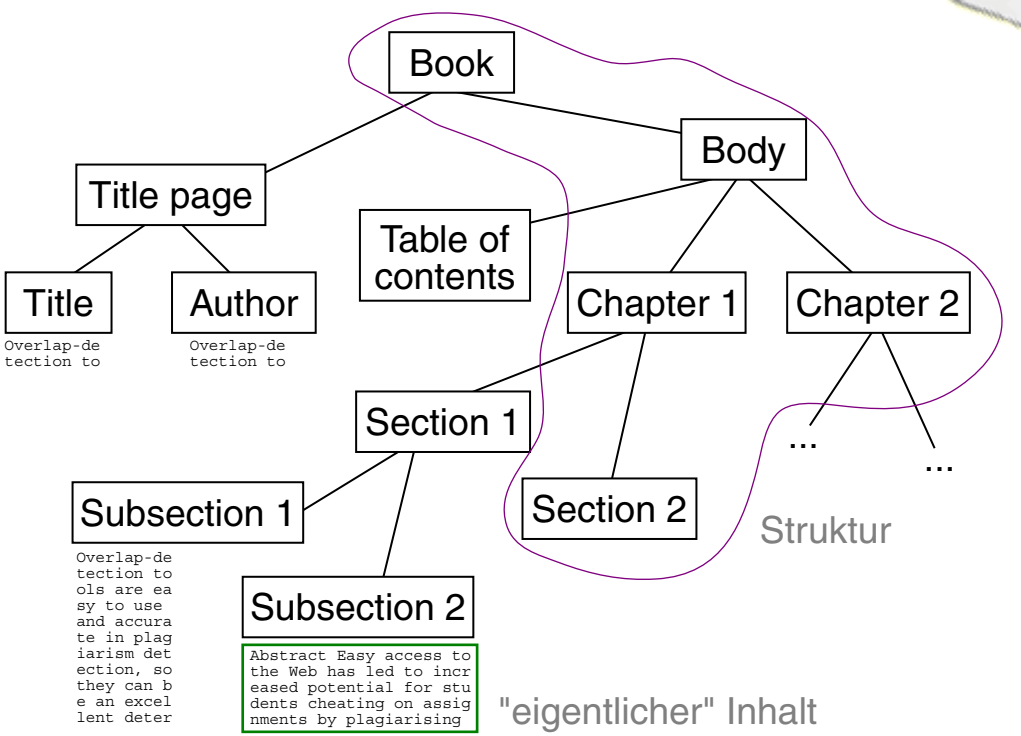
Auszeichnungssprachen

Einführung

Trennung von

1. Dokument struktur
2. Dokument inhalt
3. Dokument darstellung bzw. Layout





Darstellung

Auszeichnungssprachen

Einführung

Beispiel LaTeX:

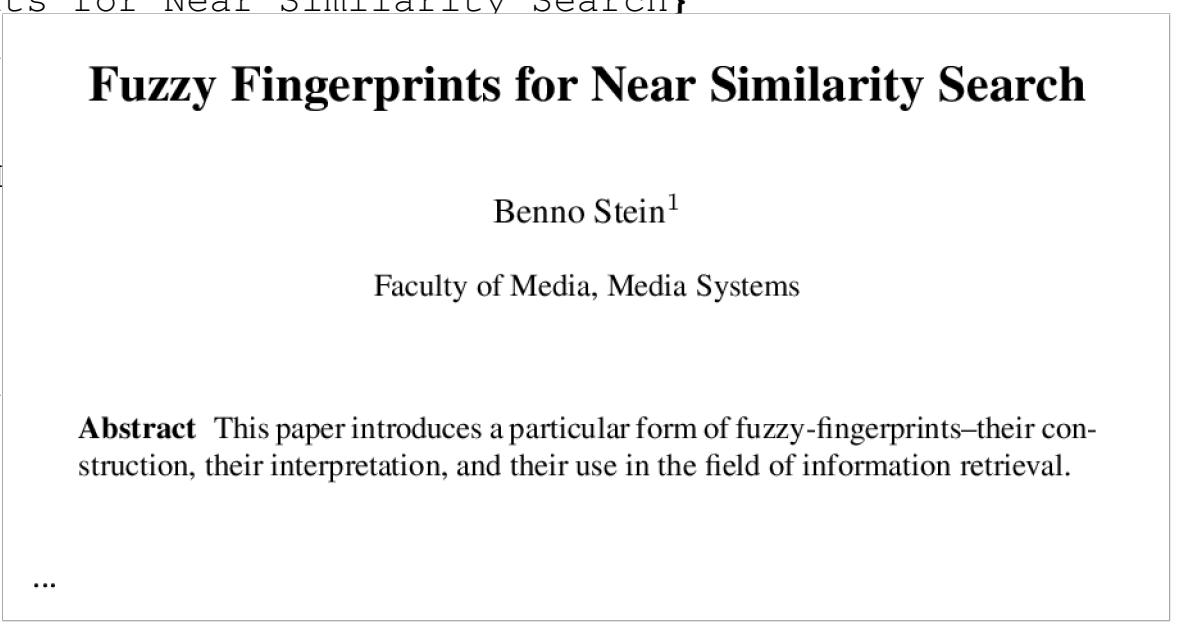
```
\documentclass{llncs}
\usepackage[T1]{fontenc}
\usepackage[german,american]{babel}
\usepackage{graphicx}

\begin{document}

\title{Fuzzy Fingerprints for Near Similarity Search}
\titlerunning{Fuzzy Fingerprints\ldots}

\author{Benno Stein\inst{1}}
\institute{Faculty of Media, Media Systems}

\maketitle

\begin{abstract}
This paper introduces a particular form of fuzzy-fingerprints--their
construction, their interpretation, and their use in the field of
information retrieval.
\end{abstract}

...
```

Auszeichnungssprachen

Einführung

Beispiel LaTeX:

```
\documentclass{llncs}
\usepackage[T1]{fontenc}
\usepackage[german,american]{babel}
\usepackage{graphicx}

\begin{document}

\title{Fuzzy Fingerprints for Near Similarity Search}
\titlerunning{Fuzzy Fin

\author{Benno Stein\ins
\institute{Faculty of M

\maketitle

\begin{abstract}
This paper introduces a
construction, their int
information retrieval.
\end{abstract}

...
```



Fuzzy Fingerprints for Near Similarity Search

Benno Stein[1]

Faculty of Media, Media Systems

Abstract This paper introduces a particular form of fuzzy-fingerprints–their construction, their interpretation, and their use in the field of information retrieval.

...

Auszeichnungssprachen

Einführung

Trennung von Struktur, Inhalt und Darstellung ermöglicht:

- Layout- und geräteunabhängige Archivierung
- maschinelle Analyse und Verarbeitung von Strukturinformation. Beispiele: Indexe, Seitenzahlen, Verweise, Zitate
- Single-Source-Prinzip: die Änderung an *einer* Quelle wird in allen nachfolgenden Layout-Prozessen nachvollzogen
- Stichworte: *Database Publishing, Cross Media Publishing*

Bemerkungen:

- Mögliche Zielformate eines Layout-Prozesses:
  - Portable Document Format PDF
  - Postscript PS
  - Rich Text Format RTF
  - Extended Markup Language XML
  - Hypertext Markup Language HTML
  - Programm-Code
  - Online-Hilfesysteme [Wikipedia]

Auszeichnungssprachen

Einführung

Merkmale von Auszeichnungssprachen:

- Strukturinformation wird in den „eigentlichen“ Inhalt integriert.
  ➜ Metasprache zur Auszeichnung von Strukturinformation
- Auszeichnung = Markup
  Auszeichnungssprache = Markup-Sprache
- Markup-Symbol *(Tag)* = Wort aus der Markup-Sprache;
  insbesondere: Unterscheidung von Start-Tags und End-Tags

Auszeichnungssprachen

Einführung

Merkmale von Auszeichnungssprachen:

- Strukturinformation wird in den „eigentlichen“ Inhalt integriert.
  ➜ Metasprache zur Auszeichnung von Strukturinformation
- Auszeichnung = Markup
  Auszeichnungssprache = Markup-Sprache
- Markup-Symbol *(Tag)* = Wort aus der Markup-Sprache;
  insbesondere: Unterscheidung von Start-Tags und End-Tags

Forderungen an Auszeichnungssprachen:

- Syntax und Semantik von Markup-Symbolen definierbar
- erweiterbar hinsichtlich neuer Strukturelemente und Dokumententypen
- von Menschen schreib- und lesbar
- einbettbar in Programmiersprachen
- offen für zukünftige Entwicklungen: neue Medientypen, Medienintegration

Auszeichnungssprachen

SGML

Historie:

- 60er einheitliches Datenformat soll Datenverarbeitung flexibler machen
- 70er C. Goldfarb entwickelt bei IBM die Generalized Markup Language GML
- 1986 Standardisierung von GML ➜ SGML = Standard GML, ISO/IEC 8879

Auszeichnungssprachen

SGML

Historie:

60er einheitliches Datenformat soll Datenverarbeitung flexibler machen

70er C. Goldfarb entwickelt bei IBM die Generalized Markup Language GML

1986 Standardisierung von GML ➜ SGML = Standard GML, ISO/IEC 8879

Konzepte von SGML:

1. SGML-Deklaration
   Definiert Zeichenvorrat, Steuerzeichen, Auszeichnungsregeln für Parser.
2. Document Type Definition, DTD (Dokumentenklasse)
   Definiert die Elementtypen eines SGML-Dokuments, „gegen“ die der Parser analysiert. Die Elementtypen bilden einen Strukturbaum.
3. SGML-Dokument
   Enthält eine Instanz des Strukturbaums gemäß einer DTD.
   Die Blätter des Strukturbaums bilden den eigentlichen Inhalt.

Auszeichnungssprachen

SGML

Historie:

60er einheitliches Datenformat soll Datenverarbeitung flexibler machen

70er C. Goldfarb entwickelt bei IBM die Generalized Markup Language GML

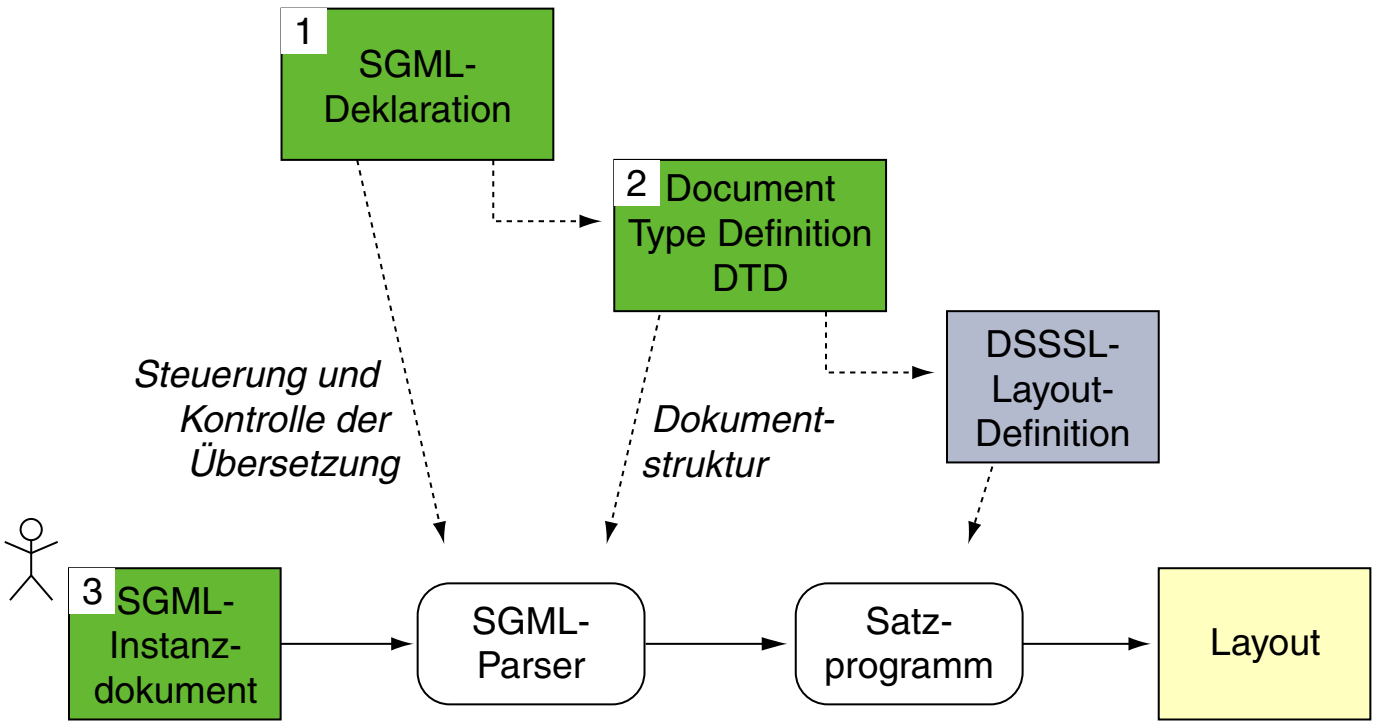
1986 Standardisierung von GML ➜ SGML = Standard GML, ISO/IEC 8879

Konzepte von SGML:

1. SGML-Deklaration
   Definiert Zeichenvorrat, Steuerzeichen, Auszeichnungsregeln für Parser.
2. Document Type Definition, DTD (Dokumentenklasse)
   Definiert die Elementtypen eines SGML-Dokuments, „gegen“ die der Parser analysiert. Die Elementtypen bilden einen Strukturbaum.
3. SGML-Dokument
   Enthält eine Instanz des Strukturbaums gemäß einer DTD.
   Die Blätter des Strukturbaums bilden den eigentlichen Inhalt.

Auszeichnungssprachen

SGML Dokumentenverarbeitung



Bemerkungen:

- Das Formatieren ist nicht Bestandteil von SGML.
- Für Layout-spezifische und geräteabhängige Definitionen zur Darstellung der in SGML beschriebenen Strukturelemente wurde eine spezielle Sprache, die *Document Style Semantics and Specification Language* DSSSL („Dissel“ ausgesprochen) entwickelt. [Wikipedia, DSSSL-Portal]
- Von Legacy-Anwendungen abgesehen wird DSSSL praktisch nicht mehr eingesetzt. Cascading Style Sheets (ab Level 3, CSS3) hat sich zu einer Alternative sowohl für DSSSL als auch für Stylesheet-Sprachen wie XSL-FO entwickelt. [Wikipedia]

Auszeichnungssprachen

SGML-Dokument

Allgemeine Form einer SGML-Elementinstanz [WT:III HTML] :

```
<elementname {attribute}*> ... </elementname>
```

Beispiel:

```
<table width="100%" cellspacing="0" cellpadding="0" border="0">
 ...
</table>
```

Auszeichnungssprachen

SGML-Dokument

Allgemeine Form einer SGML-Elementinstanz [WT:III HTML] :

```
<elementname {attribute}*> ... </elementname>
```

Beispiel:

```
<table width="100%" cellspacing="0" cellpadding="0" border="0">
 ...
</table>
```

Abstrakte Syntax für Markup-Symbole *(Tags)* [Wikipedia: BNF, EBNF] [W3C EBNF] :

```
start-tag  ::= stag-open identifier {attribute}* tag-close
end-tag    ::= etag-open identifier tag-close
attribute  ::= identifier = "value"
```

Konkrete Syntax für Markup-Symbole (festgelegt in der SGML-Deklaration):

```
stag-open  ::= <
etag-open  ::= </
tag-close  ::= >
identifier ::= {xchar}+
value      ::= {char}+
```

Auszeichnungssprachen

SGML-Dokument

Allgemeine Form einer SGML-Elementinstanz [WT:III HTML] :

```
<elementname {attribute}*> ... </elementname>
```

Beispiel:

```
<table width="100%" cellspacing="0" cellpadding="0" border="0">
 ...
</table>
```

Abstrakte Syntax für Markup-Symbole *(Tags)* [Wikipedia: BNF, EBNF] [W3C EBNF] :

```
start-tag  ::= stag-open identifier {attribute}* tag-close
end-tag    ::= etag-open identifier tag-close
attribute  ::= identifier = "value"
```

Konkrete Syntax für Markup-Symbole (festgelegt in der SGML-Deklaration):

```
stag-open  ::= <
etag-open  ::= </
tag-close  ::= >
identifier ::= {xchar}+
value      ::= {char}+
```

Auszeichnungssprachen

SGML-Dokument

Allgemeine Form einer SGML-Elementinstanz [WT:III HTML] :

```
<elementname {attribute}*> ... </elementname>
```

Beispiel:

```
<table width="100%" cellspacing="0" cellpadding="0" border="0">
 ...
</table>
```

Abstrakte Syntax für Markup-Symbole *(Tags)* [Wikipedia: BNF, EBNF] [W3C EBNF] :

```
start-tag  ::= stag-open identifier {attribute}* tag-close
end-tag    ::= etag-open identifier tag-close
attribute  ::= identifier = "value"
```

Konkrete Syntax für Markup-Symbole (festgelegt in der SGML-Deklaration):

```
stag-open  ::= <
etag-open  ::= </
tag-close  ::= >
identifier ::= {xchar}+
value      ::= {char}+
```

Auszeichnungssprachen

SGML-Dokument

Allgemeine Form einer SGML-Elementinstanz [WT:III HTML] :

```
<elementname {attribute}*> ... </elementname>
```

Beispiel:

```
<table width="100%" cellspacing="0" cellpadding="0" border="0">
 ...
</table>
```

Abstrakte Syntax für Markup-Symbole *(Tags)* [Wikipedia: BNF, EBNF] [W3C EBNF] :

```
start-tag  ::= stag-open identifier {attribute}* tag-close
end-tag    ::= etag-open identifier tag-close
attribute  ::= identifier = "value"
```

Konkrete Syntax für Markup-Symbole (festgelegt in der SGML-Deklaration):

```
stag-open  ::= <
etag-open  ::= </
tag-close  ::= >
identifier ::= {xchar}+
value      ::= {char}+
```

Auszeichnungssprachen

SGML-Dokument

Allgemeine Form einer SGML-Elementinstanz [WT:III HTML] :

```
<elementname {attribute}*> ... </elementname>
```

Beispiel:

```
<table width="100%" cellspacing="0" cellpadding="0" border="0">
 ...
</table>
```

Abstrakte Syntax für Markup-Symbole *(Tags)* [Wikipedia: BNF, EBNF] [W3C EBNF] :

```
start-tag  ::= stag-open identifier {attribute}* tag-close
end-tag    ::= etag-open identifier tag-close
attribute  ::= identifier = "value"
```

Konkrete Syntax für Markup-Symbole (festgelegt in der SGML-Deklaration):

```
stag-open  ::= <
etag-open  ::= </
tag-close  ::= >
identifier ::= {xchar}+
value      ::= {char}+
```

Auszeichnungssprachen

SGML-Dokument

Allgemeine Form einer SGML-Elementinstanz [WT:III HTML] :

```
<elementname {attribute}*> ... </elementname>
```

Beispiel:

```
<table width="100%" cellspacing="0" cellpadding="0" border="0">
 ...
</table>
```

Abstrakte Syntax für Markup-Symbole *(Tags)* [Wikipedia: BNF, EBNF] [W3C EBNF] :

```
start-tag  ::= stag-open identifier {attribute}* tag-close
end-tag    ::= etag-open identifier tag-close
attribute  ::= identifier = "value"
```

Konkrete Syntax für Markup-Symbole (festgelegt in der SGML-Deklaration):

```
stag-open  ::= <
etag-open  ::= </
tag-close  ::= >
identifier ::= {xchar}+
value      ::= {char}+
```

Auszeichnungssprachen

SGML-Dokument

Allgemeine Form einer SGML-Elementinstanz [WT:III HTML] :

```
<elementname {attribute}*> ... </elementname>
```

Beispiel:

```
<table width="100%" cellspacing="0" cellpadding="0" border="0">
 ...
</table>
```

Abstrakte Syntax für Markup-Symbole *(Tags)* [Wikipedia: BNF, EBNF] [W3C EBNF] :

```
start-tag  ::= stag-open identifier {attribute}* tag-close
end-tag    ::= etag-open identifier tag-close
attribute  ::= identifier = "value"
```

Konkrete Syntax für Markup-Symbole (festgelegt in der SGML-Deklaration):

```
stag-open  ::= <
etag-open  ::= </
tag-close  ::= >
identifier ::= {xchar}+
value      ::= {char}+
```

Auszeichnungssprachen

Document Type Definition [WT:III XML]

Die DTD definiert:

1. Art und Aufbau von Elementtypen für eine Klasse von Dokumenten
   = die Inhaltsmodelle der Elementtypen
2. die in Elementinstanzen verwendbaren Attribute und ihre Datentypen
3. verschiedene Arten von Textkonstanten, sogenannte *Entities*

Auszeichnungssprachen

Document Type Definition (Fortsetzung)

Beispiel für die Definition eines Elementtyps in einer DTD:

```
<!ELEMENT book              (titlepage, body)>
<!ELEMENT titlepage         (title, author)>
<!ELEMENT body              (table-of-contents, chapter+)>
<!ELEMENT chapter           (chapterhead, section+)>
<!ELEMENT title             (#PCDATA)>
```

Auszeichnungssprachen

Document Type Definition (Fortsetzung)

Beispiel für die Definition eines Elementtyps in einer DTD:

```
<!ELEMENT book          (titlepage, body)>
<!ELEMENT titlepage     (title, author)>
<!ELEMENT body          (table-of-contents, chapter+)>
<!ELEMENT chapter       (chapterhead, section+)>
<!ELEMENT title         (#PCDATA)>
```

`<!ELEMENT`	Beginn des Elementtyps
`chapter`	Name des Elementtyps
`(`	Beginn des Inhaltsmodells
`chapterhead,`	genau ein Kapitelkopf muss vorkommen
`section+`	mindestens ein Abschnitt muss vorkommen
`)`	Beginn des Inhaltsmodells
`>`	Ende des Elementtyps
`#PCDATA`	Datentyp “Parsed Character Data” [w3schools 1, 2]

Auszeichnungssprachen

Document Type Definition (Fortsetzung)

Beispiel für die Definition eines Elementtyps in einer DTD:

```
<!ELEMENT book              (titlepage, body)>
<!ELEMENT titlepage         (title, author)>
<!ELEMENT body              (table-of-contents, chapter+)>
<!ELEMENT chapter           (chapterhead, section+)>
<!ELEMENT title             (#PCDATA)>
```

`<!ELEMENT`	Beginn des Elementtyps
`chapter`	Name des Elementtyps
`(`	Beginn des Inhaltsmodells
`chapterhead,`	genau ein Kapitelkopf muss vorkommen
`section+`	mindestens ein Abschnitt muss vorkommen
`)`	Beginn des Inhaltsmodells
`>`	Ende des Elementtyps

`#PCDATA`	Datentyp “Parsed Character Data” [w3schools 1, 2]

Beispiele für die Definition einer Textkonstante *(Entity)*:

```
<!ENTITY euro-string        "EUR">
```

[w3schools DTD]

```
<!ENTITY euro-symbol        "&#8364;">
```
[w3schools HTML]

Bemerkungen:

- Die Elemente einer DTD können in einem SGML-Dokument instantiiert werden und dienen so im eigentlichen Inhalt als Markup.
- Entities werden durch den Aufruf &*Entityname*; referenziert.
- DTDs lassen sich auf zwei Arten einsetzen:
  1. Zur Analyse, um vorgegebene Dokumente zu validieren.
  2. Zur Synthese, um neue Dokumente zu generieren.

Auszeichnungssprachen

Zusammenhang SGML, XML, HTML, XHTML

XML, *Extensible Markup Language*, ist eine Teilmenge von SGML, die speziell auf die Bedürfnisse des WWW zugeschnitten und stark vereinfacht ist:

- XML hat eine feste, nicht veränderbare SGML-Deklaration.

Auszeichnungssprachen

Zusammenhang SGML, XML, HTML, XHTML

XML, *Extensible Markup Language*, ist eine Teilmenge von SGML, die speziell auf die Bedürfnisse des WWW zugeschnitten und stark vereinfacht ist:

- XML hat eine feste, nicht veränderbare SGML-Deklaration.

HTML, *Hypertext Markup Language*, ist eine Teilmenge von SGML und ist, verglichen mit XML, noch weiter eingeschränkt:

- HTML hat eine feste, nicht veränderbare SGML-Deklaration.
- HTML hat eine feste Dokumentstruktur und folglich nur *eine* DTD.
  ➜ Kein Austausch von SGML-Deklaration und DTD erforderlich.

Auszeichnungssprachen

Zusammenhang SGML, XML, HTML, XHTML

XML, *Extensible Markup Language*, ist eine Teilmenge von SGML, die speziell auf die Bedürfnisse des WWW zugeschnitten und stark vereinfacht ist:

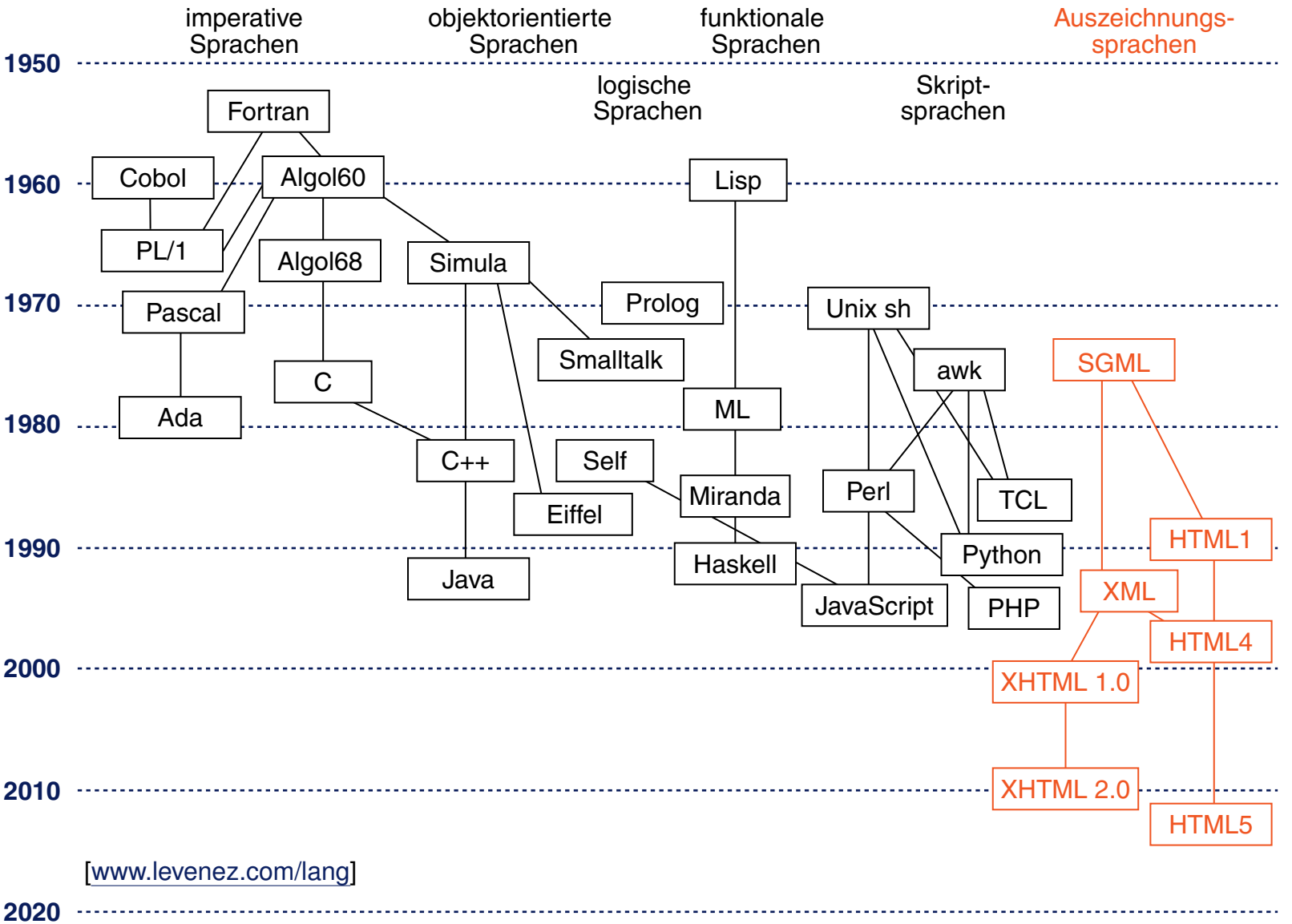
- XML hat eine feste, nicht veränderbare SGML-Deklaration.

HTML, *Hypertext Markup Language*, ist eine Teilmenge von SGML und ist, verglichen mit XML, noch weiter eingeschränkt:

- HTML hat eine feste, nicht veränderbare SGML-Deklaration.
- HTML hat eine feste Dokumentstruktur und folglich nur *eine* DTD.
  ➜ Kein Austausch von SGML-Deklaration und DTD erforderlich.

XHTML, *Extensible HyperText Markup Language*, ist die Definition von HTML auf Basis von XML.

Auszeichnungssprachen



[www.levenez.com/lang]