

## Angewandte Informatik 2 - Tutorium 2

### Besprechung: Übungsblatt 2

Götz Bürkle  
(goetz@buerkle.org)

Institut für Angewandte Informatik und  
Formale Beschreibungsverfahren - AIFB  
Universität Karlsruhe (TH)

KW 21/22

## EDIFACT Bestellung generieren

- ▶ Bestellung in „Klartext“ liegt vor
- ▶ EDIFACT Bestellung darauf generieren  
(an Aufgabe 4 von Übungsblatt 1 orientieren)
- ▶ Fehlende Informationen sinnvoll selbständig ergänzen

## Übersicht

### Übungsblatt 2

- Aufgabe 1 - EDIFACT
- Aufgabe 2 - Dokumentenbeschreibung
- Aufgabe 3 - HTML
- Aufgabe 4 - SGML
- Aufgabe 5 - HTML
- Aufgabe 6 - HTML

### Zusammenfassung

## Die Bestellung in Klartext

Bestelldatum: 22.05.2006, 11:11 Uhr

Bestellnummer: 4711

#### Käufer:

Pommes Fritz  
Auf dem Schnizel 2  
D-76128

#### Lieferant:

Schweine Park GmbH & CoKG  
Im Saustall 26-30  
D-99988 Schlachthof

Sehr geehrte Damen und Herren,  
aufgrund [...]

Pos.	Artikel-Nr.	Beschreibung	Menge
1	SAU02	Sonnenschein	50 kg
2	FERKEL33	Schäfchenwolken	45 kg
3	SAUVIEH82	Platzregen	10 kg

Bitte liefern sie termingerecht am 26.05.2006 um 17:00 Uhr.

Bei einer verspäteten Lieferung ist dieser Auftrag hinfällig. Um eine Auftragsbestätigung wird gebeten.

## Transformation - Schritte

- ▶ Was brauche ich für den Transaktions- und den Nachrichtenkopf?
- ▶ Welche Datumsangaben sind gegeben?  
In welches Format muß ich sie umwandeln?
- ▶ Was sind die Sender- und Empfängeradressen?  
Wer ist Käufer, wer Verkäufer?
- ▶ Welche Positionen hat die Bestellung?  
Wie lauten die Artikelnummern?  
Was für einer „Kategorie“ gehört ein Artikel an?
- ▶ Welche Menge steht in welcher Einheit auf der Bestellung?

## Die EDIFACT Bestellung - Teil 2

```
LIN+1++SAU02:SF' // SF für Schweinefleisch  
QTY+21:50:KG' // KG für Kilogramm  
LIN+2++FERKEL33:SF'  
QTY+21:33:KG'  
LIN+3++SAUVIEH82:SF'  
QTY+21:10:KG'  
UNT+13+1'  
UNZ+1+87654321'
```

## Die EDIFACT Bestellung - Teil 1

```
UNB+UNOA:2+3500:CODEA+4920:CODEA+060522:1111  
+87654321'  
UNH+1+ORDERS:D93A:UN'  
BGM+220+4711'  
DTM+137:20060522:1111:102'  
DTM+2:20060526:1700:102'  
NAD+BY+3500:CODEA++Pommes Fritz+  
Auf dem Schnitzel 2+Karlsruhe++76128+DE'  
NAD+SU+4920:CODEA++Schweine Park GmbH &  
CoKG+Im Saustall 26-30+Schlachthof++99988+DE'
```

## Zusammenhang XHTML, HTML

### HTML (Hypertext Markup Language)

- ▶ Auszeichnungssprache (Markup Language)
- ▶ Regeln mit Hilfe von SGML formuliert

### XHTML (Extensible Hypertext Markup Language)

- ▶ Auszeichnungssprache (Markup Language)
- ▶ Regeln mit Hilfe von XML formuliert

siehe auch in SELFHTML:

<http://de.selfhtml.org/html/xhtml/unterschiede.htm>  
<http://de.selfhtml.org/intro/technologien/html.htm#auszeichnungssprache>

Aktuelle weitergehende Informationen zu XHTML aus dem SELFHTML-Weblog:

XHTML und Schema-Validierung: <http://aktuell.de.selfhtml.org/weblog/xhtml-validierung>

## Zusammenhang XML, SGML

### XML (Extensible Markup Language)

(erweiterbare Auszeichnungs-Sprache)

- ▶ wird durch SGML (Standard Generalized Markup Language) definiert
- ▶ ist im Gegensatz zu HTML keine „SGML-Anwendung“, sondern ein „SGML-Profil“  
⇒ höhere Abstraktionsstufe (generalized markup)
- ▶ nicht in Konkurrenz zu HTML, sondern eher zu SGML
- ▶ wie mit SGML kann man mit XML eigene (Auszeichnungs-)Sprachen definieren
- ▶ XML-Grundkonventionen müssen eingehalten werden

## XML, Sprachmächtigkeit

### → XML

- ▶ Bestandteile einer Sprache legt bei XML meist ein Schema fest
- ▶ im Schema kann/sollte genau festgelegt werden, welche Elemente innerhalb welcher Elemente stehen dürfen und welche Elemente welche Attribute haben dürfen oder müssen
- ▶ XHTML ist eine mit XML definierte Sprache

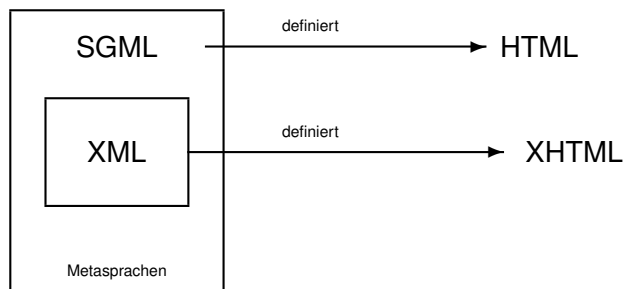
siehe auch <http://de.selfhtml.org/intro/technologien/xml.htm#definitionssprache>

## Sprachmächtigkeit

HTML  $\in$  SGML

XHTML  $\in$  XML  $\subset$  SGML

anschaulich:



## Dokumenttypdeklarationen - Bedeutung der DTD

- ▶ jede gültige (valide) HTML Datei muß die in einer DTD definierten Regeln erfüllen
- ▶ DTD legt fest, welche Elemente ein (HTML-)Dokument enthalten darf, und wie diese ineinander verschachtelt sein dürfen. Außerdem, welche Attribute welches Element haben darf oder auch haben muß.

## Dokumenttypdeklarationen - Aufbau einer DTD

```
<!DOCTYPE HTML PUBLIC
"-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

- ▶ **DTD muß vor dem einleitenden <html>-Tag stehen**
- ▶ **DOCTYPE HTML PUBLIC:** Deklaration bezieht sich auf öffentlich verfügbare HTML-DTD
- ▶ **W3C:** Herausgeber der DTD, hier das World Wide Web Consortium, kurz W3C
- ▶ **DTD HTML 4.01 Transitional:** Es wird der Dokumenttyp HTML Version 4.01 in der Version Transitional verwendet

## Dokumenttypdeklarationen - unterschiedliche DTDs

- ▶ **Strict:** Restriktivste DTD, einige Elemente (z.B. <font>) sind nicht erlaubt. Textformatierung mit CSS (Cascading Style Sheets) ist vorgeschrieben. Text im <body> muß innerhalb anderer Elemente stehen.
- ▶ **Transitional:** Erlaubt viele, was bei Strict verboten ist (z.B. target-Attribut für Verweise).
- ▶ **Frameset:** Nur für Dokumente, in denen Framesets definiert werden.

## Dokumenttypdeklarationen - Aufbau einer DTD

```
<!DOCTYPE HTML PUBLIC
"-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

- ▶ **EN:** Sprachkürzel, legt die Sprache der Element- und Attributnamen fest.
- ▶ **http://www.w3.org/TR/html4/loose.dtd:** Diese Angabe ist optional, sollte jedoch angegeben werden.

siehe auch <http://de.selfhtml.org/html/allgemein/grundgeruest.htm#dokumenttyp>

## MiniHTML anpassen

Aufgabe ist die in der Vorlesung behandelte DTD für MiniHTML wie folgt zu erweitern:

- ▶ Element für **Unterstrichungen** (Elementname: UNTERSTRICH)
- ▶ Element für **KAPITÄLCHEN** (Elementname: KAPS)

## MiniHTML-DTD aus der Vorlesung

```
<!DOCTYPE MiniHTML [  
  <!ELEMENT MiniHTML O O (H1|P)* >  
  <!ELEMENT (H1 | P) - -  
  (#PCDATA|HERVOR|BETONT)* >  
  <!ELEMENT (HERVOR|BETONT) - - (#PCDATA) >  
>
```

## Beispiel in MiniHTML

```
<MiniHTML>  
  <H1>Eine Überschrift.</H1>  
  <P>Ein Absatz.</P>  
  <P>Eine <HERVOR>Hervorhebung</HERVOR>.</P>  
  <P>Etwas <BETONT>fett  
Gedrucktes</BETONT>.</P>  
  <P>Etwas  
<UNTERSTRICH>Unterstrichenes</UNTERSTRICH> und  
sogar etwas in  
<KAPS>Grossbuchstaben</KAPS>.</P>  
</MiniHTML>
```

## angepaßte MiniHTML DTD

```
<!DOCTYPE MiniHTML [  
  <!ELEMENT MiniHTML O O (H1|P)* >  
  <!ELEMENT (H1|P) - -  
  (#PCDATA|HERVOR|BETONT|UNTERSTRICH|KAPS)* >  
  <!ELEMENT (HERVOR|BETONT|UNTERSTRICH|KAPS) -  
  - (#PCDATA) >  
>
```

Die „-“ bzw. „O“ bedeuten, daß Beginn- und Endtag zwingend vorgeschrieben sind bzw. ausgelassen (omitted) werden können.

## Ergänzungen in der style-sheet.dtd

```
(element UNTERSTRICH (make sequence  
  font-decoration : 'underline))  
(element KAPS (make sequence  
  font-variant : 'small-caps))
```

## HTML-Code angeben

Einmal	nach	rechts
	und	
zurück		

- ▶  $a^3 + b^3 \neq c^3$
- ▶ **Manchmal stimmt es, mal nicht.**
- ▶ Der dritte Punkt.

Viel Spaß bei der Bearbeitung!

## HTML-Code - Body: Tabelle Teil 1

```
<!DOCTYPE ...  
<html>  
  <head> ... </head>  
  <body style="font-family:Arial,Sans Serif;  
color:#000000;">  
    <table border="0" align="center"  
bgcolor="#CCCCCC">  
      <tr>  
        <td>Einmal</td>  
        <td>&nbsp;</td>  
        <td>&nbsp;</td>  
      </tr>  
      ...  
    </table>  
    ...  
</body>  
</html>
```

## HTML-Code - Head

```
<!DOCTYPE HTML PUBLIC  
  "-//W3C//DTD HTML 4.01 Transitional//EN"  
  "http://www.w3.org/TR/html4/loose.dtd">  
<html>  
  <head>  
    <meta http-equiv="content-type"  
content="text/html; charset=ISO-8859-1">  
    <title>Aufgabe 5</title>  
  </head>  
  <body ...>  
    ...  
  </body>  
</html>
```

## HTML-Code - Body: Tabelle Teil 2

```
<!DOCTYPE ...  
<html>  
  <head> ... </head>  
  <body ...>  
    <table ...>  
      ...  
      <tr>  
        <td>&nbsp;</td>  
        <td>nach</td>  
        <td>&nbsp;</td>  
      </tr>  
      ...  
    </table>  
    ...  
</body>  
</html>
```

## HTML-Code - Body: Tabelle Teil 3

```
<!DOCTYPE ...  
<html>  
  <head> ... </head>  
  <body ...>  
    <table ...>  
      ...  
      <tr>  
        <td>&nbsp;</td>  
        <td>&nbsp;</td>  
        <td>rechts</td>  
      </tr>  
      ...  
    </table>  
    ...  
  </body>  
</html>
```

## HTML-Code - Body: Tabelle Teil 5

```
<!DOCTYPE ...  
<html>  
  <head> ... </head>  
  <body ...>  
    <table ...>  
      ...  
      <tr>  
        <td>zur&uuml;ck</td>  
        <td>&nbsp;</td>  
        <td>&nbsp;</td>  
      </tr>  
    </table>  
    ...  
  </body>  
</html>
```

## HTML-Code - Body: Tabelle Teil 4

```
<!DOCTYPE ...  
<html>  
  <head> ... </head>  
  <body ...>  
    <table ...>  
      ...  
      <tr>  
        <td>&nbsp;</td>  
        <td>und</td>  
        <td>&nbsp;</td>  
      </tr>  
      ...  
    </table>  
    ...  
  </body>  
</html>
```

## HTML-Code - Body: Liste Teil 1

```
<!DOCTYPE ...  
<html>  
  <head> ... </head>  
  <body ...>  
    ...  
    <ul>  
      <li style="font-family:Times,Serif">  
        <i>a</i><sup>3</sup> + <i>b</i><sup>3</sup>  
        &ne; <i>c</i><sup>3</sup>  
      </li>  
      ...  
    </ul>  
    ...  
  </body>  
</html>
```

## HTML-Code - Body: Liste Teil 2

```
<!DOCTYPE ...  
<html>  
  <head> ... </head>  
  <body ...>  
    ...  
    <ul>  
      ...  
      <li>  
        <strong>Manchmal </strong>  
        <strong><u>stimmt es</u></strong>,&br/>        <u>mal</u> nicht.</li>  
      <li>Der dr&icirc;tt&eacute; Punkt.</li>  
    </ul>  
    ...  
  </body>  
</html>
```

## HTML-Fehlerjagd

Findet die Fehler und erstellt ein **valides Dokument zum aktuellen HTML-Standard 4.01**.

Fehlerhafter Quelltext  
→ siehe Übungsblatt!

## HTML-Code - Body: Schlußsatz

```
<!DOCTYPE ...  
<html>  
  <head> ... </head>  
  <body ...>  
    ...  
    <p></p>  
    <p align="right">  
      Viel <big>Spa&szlig;</big> bei der  
      <small>Bearbeitung</small>!  
    </p>  
  </body>  
</html>
```

## HTML-Fehlerjagd - Fehler

- ▶ Fehlendes `<HTML>`-Tag am Anfang und am Ende
- ▶ Fehlerhaftes Kommentar-Tag
- ▶ Ungültiger Attributsname beim `<body>`-Tag
- ▶ Das `<H1>`-Tag wird zweimal geöffnet statt einmal geschlossen
- ▶ Im ersten Absatz ist das `<p>`-Tag vergessen worden
- ▶ Anstatt eines ü sollte man `&uuml;` schreiben

## HTML-Fehlerjagd - Fehler

- ▶ Beim 3. Aufzählungselement wird das `<font>`-Tag 4x geöffnet und nur 3x geschlossen. Grundsätzlich sollte man in HTML 4.01 auf das `<font>`-Tag verzichten und mit CSS arbeiten!
- ▶ Das `<break>`-Tag ist falsch. Es muss `<br>` lauten
- ▶ Es fehlt das schließende `<p>`-Tag

## Zusammenfassung

- ▶ **EDIFACT**: Aus Klartextbestellung EDIFACT-Bestellung erzeugen können
- ▶ **Dokumentenbeschreibung**: HTML, XHTML, XML und SGML gegeneinander abgrenzen können
- ▶ **SGML**: Eine SGML-DTD lesen, verstehen und erweitern können
- ▶ **HTML**: HTML-Dokumente selbst erstellen und in HTML-Dokumenten Syntaxfehler finden und korrigieren können

## HTML-Fehlerjagd - Fehler

- ▶ In der Tabelle fehlen alle schließenden `<td>`-Tags sowie alle schließenden `<tr>`-Tags. Die inkonsistente Schreibweise des schließenden `<table>`-Tags ist erlaubt!
- ▶ Das schließende `<center>`-Tag fehlt. Grundsätzlich sollte man in HTML 4.01 auf das `<center>`-Tag verzichten und stattdessen mit `<div>`-Tags arbeiten!
- ▶ Das schließende `<body>`-Tag fehlt
- ▶ ... (es waren mit Sicherheit noch mehr Fehler in der Datei)