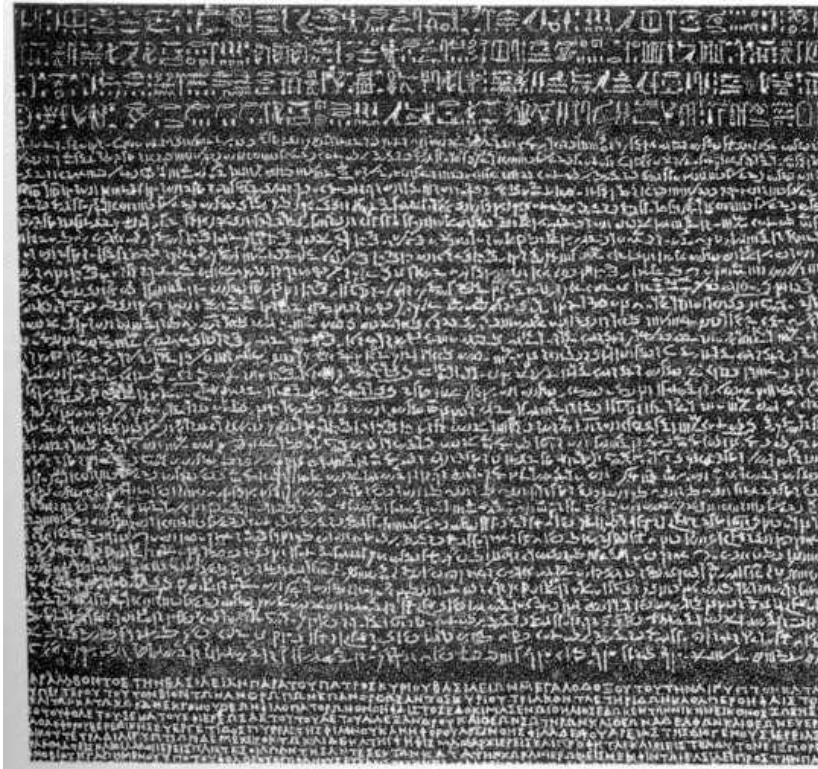


Text



Text: XML

1. Markupsprachen: XML (EXtensible Markup Language)

1.1 Charakteristik

- XML ist ein reines Text-Format. XML-Dateien beinhalten ausschließlich ASCII bzw. Unicodezeichen und können in jedem Texteditor betrachtet und bearbeitet werden.
- XML besteht grundsätzlich aus 3 Elementen, die unabhängig voneinander existieren können und innerhalb des XML-Dokuments eigenständige Einheiten bilden:
 - Die **Daten** oder der **Textinhalt** werden von Tags geklammert, d.h. von einem Start-Tag und einem End-Tag umschlossen.
 - die **Struktur** des Textes kann z.B. über *DTDs* (Document Type Definition) und *XML Schema* (XSD: XML Schema Definition) beschrieben werden.
 - die **Form und Darstellung** des Textes kann über *CSS* (Cascading Style Sheets) und *XSL* (eXtensible Stylesheet Language) festgelegt werden.
- Über die Verwendung von XML können also **Inhalt (Daten)**, **Struktur** und **Layout** eines Dokuments strikt getrennt werden.

Text: XML

1. Markupsprachen: XML (EXtensible Markup Language)

1.2 Wohlgeformte und gültige Dokumente

- XML-Dokumente *müssen* **wohlgeformt** (well-formed) sein, d.h. sie müssen den Regeln der XML-Syntax genügen (hier ein kleiner Auszug der Syntax-Regeln):
 - Jedes Start-Tag muß ein dazugehöriges End-Tag haben.
 - Elemente dürfen geschachtelt sein, sich aber nicht überlappen.
 - Es muß genau ein Wurzelement geben.
 - Attributwerte müssen in Anführungszeichen stehen.
 - Ein Element darf nicht zwei Attribute mit dem gleichen Namen besitzen.
 - Kommentare und Verarbeitungsanweisungen dürfen nicht innerhalb von Tags stehen.
 - In den Zeichendaten eines Elementes oder Attributes dürfen keine ungeschützten ,<, oder ,&'-Zeichen stehen.
- XML-Dokumente *können* **gültig** sein. **Gültige (valide) Dokumente** müssen strengeren Anforderungen bzgl. Ihrer Struktur genügen.
- Diese Struktur kann entweder in einer **DTD (Document Type Definition)** oder in einer **XML-Schema-Datei** beschrieben werden.
- DTDs sind hierbei der ursprüngliche Weg, XML-Schema ist die neuere, mächtigere Lösung, den Aufbau von XML-Dokumenten zu beschreiben.
- Ein weiterer Vorteil liegt darin, dass DTDs eine eigene Syntax haben, XML-Schema ist eine XML-Anwendung und nutzt die bekannte XML-Syntax.

Text: XML

- Zunächst macht es scheinbar mehr Arbeit, **gültige** Dokumente zu erzeugen: Man muss zunächst die Struktur des Dokumentes vollständig in einer DTD oder einem XML-Schema definieren und dann das Dokument selbst erstellen, wobei dieses dann allen Spezifikationen der DTD oder des Schemas genügen muss. Warum sollte man sich also die Mühe machen?
- DTDs oder **Schemas** erlauben es, Dokumente bzgl. Ihrer Struktur zu prüfen.
- **Gültige Dokumente** sind nützlich, wenn die Einheitlichkeit einer Gruppe gleichartiger Dokumente gewährleistet werden muss.
- Dies ist z.B. der Fall, wenn XML-Dokumente maschinell von einer Software-Applikation verarbeitet werden sollen. Ist das Dokument **gültig**, kann es auch mit Sicherheit von der Software gelesen werden.

1.3 Struktur: DTD und XML-Schema

1.3.1 Vergleich

	DTD	XML Schema
Syntax	eigene	XML konform
Datentypen	string	Vordefinierte Datentypen, auf deren Basis Erweiterungen(eigene Definitionen vorgenommen werden können; Wertezuweisung
Komplexität	Einfach, intuitiv	Hoch; z.B. komplexe Typen
Verarbeitbarkeit	-	+
Namensräume	Alle NS müssen für ein valides XML-Dokument in einer DTD definiert werden	Unterstützt Angabe verschiedener NS

1.3.2 Vergleich Aufbau: DTD und XML Schema

Bsp.: artikel.xml

Bsp. mit DTD-Einbindung:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE artikel SYSTEM "artikel.dtd">
<artikel>
<titel>Der Titel</titel>
<inhalt>Der Inhalt</inhalt>
</artikel>
```

Bsp. Mit XML Schema-Einbindung

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<artikel xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://www.hki.uni-koeln.de/artikel.xsd">
<titel>Der Titel</titel>
<inhalt>Der Inhalt</inhalt>
</artikel>
```

- Eine **Schemadatei** wird über das **Wurzelement** in die Instanzdatei eingebunden.
- Über das Attribut **xmlns:xsi** deklariert man eine Instanz eines Namensraums, über (z.B.) **xsi:noNamespaceSchemaLocation** wird der Ort/Name der Schemadatei referenziert.

1.3.3 XML Schema: Aufbau

artikel.dtd

```
<!ELEMENT artikel (titel, inhalt )>  
<!ELEMENT titel (#PCDATA)>  
<!ELEMENT inhalt (#PCDATA)>
```

artikel.xsd

```
<?xml version="1.0"encoding="ISO-8859-1"?>  
  <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">  
    <xs:element name="artikel">  
      <xs:complexType>  
        <xs:sequence>  
          <xs:element name="titel" type="xs:string"/>  
          <xs:element name="inhalt" type="xs:string"/>  
        </xs:sequence>  
      </xs:complexType>  
    </xs:element>  
  </xs:schema>
```

artikel.xsd

```
<?xml version="1.0" encoding="ISO-8859-1"?>
  <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="artikel">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="titel" type="xs:string" maxOccurs="1"/>
          <xs:element name="inhalt" type="xs:string"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:schema>
```

- Elemente werden in XML Schema durch **xs:element** benannt. Das **Präfix** wird im Wurzelement **xs:schema** festgelegt, das auch den Verweis auf den Namespace enthält. (Die Zeile `xmlns:xsd` würde zur Folge haben, dass alle weiteren Tags das Präfix `xsd:` tragen.)
- Das Tag Element enthält idR die Attribute **name und type** .
- Es kann optional **weitere Attribute** enthalten. Das Attribut `type` erhält ebenfalls das Präfix.

artikel.xsd

```
<?xml version="1.0" encoding="ISO-8859-1"?>
  <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="artikel">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="titel" type="xs:string" maxOccurs="1"/>
          <xs:element name="inhalt" type="xs:string"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:schema>
```

- Die Verschachtelung von Elementen wird in XML Schema über das Tag mit der Benennung **complexType** umgesetzt.
- Ihm folgt ein weiteres Tag **sequence**, das die einzelnen Elemente auf einer Ebene aufnimmt.

Text: XML

1.4 Aussehen und Form der Ausgabe einer XML-Datei

- Will man eine XML-Datei weiterverarbeiten und deren Inhalte zur Ausgabe aufbereiten, kann man sich u.a. *XSL (eXtensible Stylesheet Language)* bedienen.
- **XSL** ist eine Familie von Sprachen zur Erzeugung druckbarer Layouts aus XML-Dokumenten.
- Zu **XSL** gehören
 - das XML-basierte eigentliche **XSL** (zur Unterscheidung genannt **XSL Formatting Objects** oder **XSL-FO**).
EXKURS: XSL-FO ist eine XML-Anwendung zur Beschreibung der genauen Anordnung von Text auf einer Seite. Sie besitzt Elemente, die Seiten, Textblöcke auf den Seiten, Grafiken und horizontale Linien und anderes repräsentieren. Meist werden Sie jedoch XSL-FO nicht direkt schreiben. Statt dessen schreiben Sie ein XSLT-Stylesheet, das die Auszeichnungen ihres Dokuments in XSL-FO umwandelt. Die Anwendung, die das Dokument für die Anzeige aufbereitet, liest das XSL-FO und gibt es für die Benutzer aus. Da keiner der großen Browser momentan die direkte Aufbereitung von XSL-FO-Dokumenten unterstützt, gibt es normalerweise einen dritten Schritt, in dem das XSL-FO in ein drittes Format, wie PDF oder T_EX umgewandelt wird.
 - das XML-basierte XSLT (XSL Transformation) für die Transformation eines beliebigen XML-Dokuments in ein anderes Format, wie z.B. Text oder HTML, anhand bestimmter Regeln.
 - und XPath für die Adressierung von Elementen eines XML-Dokuments.

XSLT-Beispiel: Das XML-Dokument

```
<?xml version="1.0"?>
```

```
<!-- Dateiname: XsltDemo01.xml -->
```

```
<?xml-stylesheet type="text/xsl" href="XsltDemo01.xsl"?>
```

```
<BUCH>
```

```
  <TITEL>Moby-Dick</TITEL>
```

```
  <AUTOR>
```

```
    <VORNAME>Herman</VORNAME>
```

```
    <NACHNAME>Melville</NACHNAME>
```

```
  </AUTOR>
```

```
  <EINBAND>Gebundene Ausgabe</EINBAND>
```

```
  <SEITEN>724</SEITEN>
```

```
  <PREIS>9,95</PREIS>
```

```
</BUCH>
```

XSLT-Beispiel: Das XSLT-Stylesheet

```
<?xml version="1.0"?>

<!-- Dateiname: XsltDemo01.xsl -->

<xsl:stylesheet
  version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="/"> <!-- den XSLT-Wurzelknoten zuordnen -->
    <HTML>
      <HEAD>
        <TITLE>Buchbeschreibung</TITLE>
      </HEAD>
      <BODY>
        <H2>Buchbeschreibung</H2>
        <SPAN STYLE="font-style:italic">Autor: </SPAN>
          <xsl:value-of select="BUCH/AUTOR"/><BR/>
        <SPAN STYLE="font-style:italic">Titel: </SPAN>
          <xsl:value-of select="BUCH/TITEL"/><BR/>
        <SPAN STYLE="font-style:italic">Preis: </SPAN>
          <xsl:value-of select="BUCH/PREIS"/><BR/>
        <SPAN STYLE="font-style:italic">Einband: </SPAN>
          <xsl:value-of select="BUCH/EINBAND"/><BR/>
        <SPAN STYLE="font-style:italic">Seitenzahl: </SPAN>
          <xsl:value-of select="BUCH/SEITEN"/>
      </BODY>
    </HTML>
  </xsl:template>
</xsl:stylesheet>
```

XSLT-Beispiel: Die Ausgabe im Browser



Text: RTF

2. Textverarbeitungsformate: RTF (Rich Text Format)

- RTF wurde Mitte der 80er Jahre durch Microsoft definiert, um Text und Grafiken zwischen verschiedenen Anwendungen zu transferieren.
- Damit das Format auf verschiedenen Plattformen benutzbar ist, werden nur anzeigbare Zeichen der ANSI-, MAC- und PC-Zeichensätze verwendet, um Texte und Formatinformationen zu speichern.
- Jede RTF-Datei besteht aus
 - *unformatiertem Text*,
 - *Kontrollwörtern* und
 - *Kontrollzeichen*, die zu
 - *Gruppen* zusammen gefasst werden.

DIES IST EIN WORDPAD DOKUMENT

Fette Schrift

Kursive Schrift

Anderer Schrifttyp

Größere Schrift

Farbe

noch eine Farbe

```
{\rtf1\ansi\ansicpg1252\deff0\deflang1031{\fonttbl{\f0\fswiss\fcharset0  
Arial;}{\f1\fmodern\fprq1\fcharset0 Courier New;}{\f2\fswiss\fprq2\fcharset0 Arial;}}  
{\colortbl ;\red255\green0\blue0;\red0\green0\blue255;}  
\viewkind4\uc1\pard\f0\fs20 DIES IST EIN WORDPAD DOKUMENT\par  
\par  
\b Fette Schrift\par  
\b0\i Kursive Schrift\par  
\i0\f1 Anderer Schrifttyp\par  
\f2\fs36 Gr'\f6'\dfere Schrift\par  
\cf1\fs32 Farbe\f0\fs20\par  
\cf2\f2\fs32 noch eine Farbe\par  
\par  
\cf1\f0\fs20\par  
}
```

Text: RTF

2.1 RTF-Kontrollwörter

- Ein Kontrollwort beginnt mit einem Backslash, gefolgt von einem Schlüsselwort und endet mit einem Delimiter (Begrenzer).
Also: `\Schlüsselwort <delimiter>`
- Das Schlüsselwort darf nur Kleinbuchstaben enthalten (RTF ist case-sensitive).
- Der Delimiter darf aus folgenden Zeichen bestehen:
 - Leerzeichen
 - Ziffer oder –
 - alle anderen Zeichen mit Ausnahme der Buchstaben

```
{\rtf1\ansi\ansicpg1252\deff0\deflang1031{\fonttbl{\f0\fswiss\fcharset0  
Arial;}{\f1\fmodern\fprq1\fcharset0 Courier New;}{\f2\fswiss\fprq2\fcharset0  
Arial;}}  
{\colortbl ;\red255\green0\blue0;\red0\green0\blue255;}  
\viewkind4\uc1\pard\f0\fs20 DIES IST EIN WORDPAD DOKUMENT\par  
\par ....
```

Text: RTF

2.2 RTF-Steuersymbole

- Ein Steuersymbol beginnt mit einem Backslash \, gefolgt von einem weiteren Zeichen (nicht aus dem Alphabet).

Also: \ <Zeichen>

```
{\rtf1\ansi\ansicpg1252\deff0\deflang1031{\*\fonttbl{\f0\fswiss\fchars  
et0 Arial;}{\f1\fmodern\fprq1\fcharset0 Courier  
New;}{\f2\fswiss\fprq2\fcharset0 Arial;}}
```

(* =Kontrollwörter, die erst in späteren Spezifikationen hinzugefügt wurden)

Text: RTF

2.3 RTF-Gruppe

- Eine **Gruppe** besteht aus **Text**, sowie **Kontroll-** und **Steuerwörtern**, die in geschweifte Klammern **{ }** eingeschlossen werden.
Also: { =Anfang der Gruppe
 } =Ende der Gruppe
- Jede Gruppe spezifiziert den Text, der durch die Gruppe formatiert oder sonst wie beeinflusst wird.
- Eine RTF-Datei kann zusätzliche Gruppen mit Definitionen von Fonts, Bildschirmfarben, Bildern, Fußnoten etc. enthalten.
- Diese Informationen müssen vor dem ersten Text auftreten.

```
{\rtf1\ansi\ansicpg1252\deff0\deflang1031{\fonttbl{\f0\fswiss\fcharset0  
Arial;}{\f1\modern\prq1\fcharset0 Courier New;}{\f2\fswiss\prq2\fcharset0  
Arial;}}  
{\colortbl ;\red255\green0\blue0;\red0\green0\blue255;}  
\viewkind4\uc1\pard\f0\fs20 DIES IST EIN WORDPAD DOKUMENT\par  
\par ....
```

Text: RTF

2.4 Verwendung von Kontrollwörtern

- Tritt eines der Zeichen `\ { }` im Text auf, muss es durch einen Backslash *maskiert* werden.
- Einige Kontrollwörter schalten eine Eigenschaft ein.
Z.B.: `\b` für Fettdruck
- In RTF 1.0: Um die Eigenschaft auszuschalten, wird die Ziffer 0 an das Kontrollwort angehängt:
Z.B.: `\b0` für Ende Fettdruck
- In RTF 1.6: Alles was in der Gruppe steht unterliegt den Eigenschaften.
- Kontrollwörter, die mit der Zeichenfolge `*` beginnen, waren in der ersten RTF-Spezifikation nicht enthalten und sind später hinzugefügt worden.

2.5

Der RTF-Header

- Der **Headerbereich** enthält die Gruppen
 - RTF-Versionsnummer
 - Zeichensatzvereinbarungen
 - Fontbeschreibungen

```
{\rtf1\ansi\ansicpg1252\deff0\deflang1031{\fonttbl{\f0\fswiss\fcharset0
Arial;}{\f1\fmmodern\fprq1\fcharset0 Courier
New;}{\f2\fswiss\fprq2\fcharset0 Arial;}}
{\colortbl ;\red255\green0\blue0;\red0\green0\blue255;}
\viewkind4\uc1\pard\fs20 DIES IST EIN WORDPAD DOKUMENT\par
\par
\lb Fette Schrift\par
\lb0\li Kursive Schrift\par
\li0\l1 Anderer Schrifttyp\par
\l2\fs36 Gr'\f6\dfere Schrift\par
\lcf1\fs32 Farbe\fs20\par
\lcf2\l2\fs32 noch eine Farbe\par
\par
\lcf1\fs20\par
}
```

2.5.1 RTF-Versionsnummer

- Als erstes wird mit `\rtfN` die Versionsnummer der RTF-Spezifikation angegeben. **N** steht für die Versionsnummer.

Also: `\rtf <parameter>`

2.5.2 RTF-Zeichensatzvereinbarungen

- Als zweites ist der verwendete Zeichensatz zu vereinbaren.
 - `\ansi` = ANSI (Standard)
 - `\mac` = Apple Macintosh
 - `\pc` = IBM PC mit Codepage 437
 - `\pca` = IBM PC mit Codepage 850
 - In Word 2000 lassen sich Texte in Unicode hinterlegen, deshalb wurde in RTF eine Erweiterung zur Unterstützung dieser Unicode-Zeichen eingeführt:
 - `\ansicpgN`
 - =eine ANSI-Codeseite *N*, die zur Konvertierung des Unicode/ANSI-Zeichensatzes herangezogen wird.
 - Standardmäßig wird der ANSI-Code der Betriebssystemumgebung benutzt. `\ansicpg1252` = Western European

```
{\rtf1\ansi\ansicpg1252\deff0\deflang1031{\fonttbl{\f0\fswiss\fcharset0
Arial;}{\f1\fmodern\fprq1\fcharset0 Courier
New;}{\f2\fswiss\fprq2\fcharset0 Arial;}}
{\colortbl ;\red255\green0\blue0;\red0\green0\blue255;}
\viewkind4\uc1\pard\f0\fs20 ...
```

Text: RTF

2.5.3 RTF-Fontbeschreibungen

- Des weiteren werden die verwendeten Fonts beschrieben.
 - \fN
= weist jedem verwendeten Font eine eindeutige Nummer zu
 - \fontfamilie
 - \zeichensatz

```
{\rtf1\ansil\ansicpg1252\deff0\deflang1031{\fonttbl{\f0\fswiss\fcharset0
Arial;}{\f1\fmodern\fprq1\fcharset0 Courier
New;}{\f2\fswiss\fprq2\fcharset0 Arial;}}
{\colortbl ;\red255\green0\blue0;\red0\green0\blue255;}
\viewkind4\uc1\pard\f0\fs20 DIES IST EIN WORDPAD DOKUMENT\par
\par
\lFette Schrift\par
\lKursive Schrift\par
\l\i\l1 Anderer Schrifttyp\par
\l\l2\fs36 Gr\l'f6\l'dfere Schrift\par
\l\cf1\fs32 Farbe\l'f0\fs20\par
\l\cf2\l'f2\fs32 noch eine Farbe\par
\par
\l\cf1\l'f0\fs20\par
}
```

2.6 Der RTF-Dokumentbereich

- Der Dokumentbereich enthält die Gruppen
 - Info
 - Dokumentformat
 - Abschnitt
- Info ist optional, die Gruppen Dokumentformat und Abschnitte können mehrfach wiederholt werden.

```
{\rtf1\ansi\ansicpg1252\deff0\deflang1031
{\fonttbl{\f0\fswiss\fcharset0 Arial;}
{\f1\modern\fprq1\fcharset0 Courier New;}
{\f2\fswiss\fprq2\fcharset0 Arial;}}
{\colortbl ;\red255\green0\blue0;\red0\green0\blue255;}
{\info{\title DIES IST EIN MSWORD DOKUMENT}
{\author herrmann}{\operator herrmann}
{\creatim\yr2005\mo5\dy11\hr21\min2}{\revtim\yr2005\mo5\dy11\hr21\min6}
{\version1}{\edmins0}{\nofpages1}{\nofwords13}{\nofchars87}
{\nofcharsws99}{\vern16437}}
\paperw11906\paperh16838\margl1417\marginr1417\margt1417\marginb1134
\deftab708\widowctrl\ftnbj\aeenddoc\hyphhotz425\noxlattoyen\expshrt\tr\lspc
\dn\tblnsbdb\nospaceforu\ \fet0\sect \linex0\headery708\footery708\colsx708
```

...

Text: RTF

2.7 Die Informationsgruppe

- Die Gruppe der Dokumentinformationen wird durch das Kontrollwort `\info` eingeleitet.
- Mögliche Informationen sind u.a.:
 - Titel des Dokuments (`\title`)
 - Autor (`\author`)
 - Interne Versionsnummer (`\vernM`)
 - Zeitangaben (`\yrN\moN\dyN\hrN\minN`)
 - Zeitpunkt der Dokumentanlage (`\creatim`)
 - Zeitpunkt der letzten Änderung (`\revtim`)
 - Zeitpunkt des letzten Ausdrucks (`\printim`)
 - addierte Zeit zur Bearbeitung des Dokuments (`\edminsM`)
 - Quantifizierende Angaben
 - Anzahl der Seiten (`\nofpagesM`)
 - Anzahl der Wörter (`\nofwordsM`)
 - Anzahl der Zeichen (`\nofcharsM`)

```
{
```

```
\info
```

```
{\title DIES IST EIN MSWORD DOKUMENT}
```

```
{\author herrmann}
```

```
{\operator herrmann}
```

```
{\creatim\yr2005\mo5\dy11\hr21\min2}
```

```
{\revtim\yr2005\mo5\dy11\hr21\min6}
```

```
{\version1}
```

```
{\edmins0}
```

```
{\nofpages1}
```

```
{\nofwords13}
```

```
{\nofchars87}
```

```
{\nofcharsws99}
```

```
{\vern16437}
```

```
}
```

Text: RTF

2.8 Das Dokumentformat

- Die Kontrollwörter des Dokumentformates definieren die *Eigenschaften* (Ränder, Fußnotenpositionen, etc.) des Dokuments.
- Die Kontrollwörter stehen nach der Gruppe \info, aber vor dem eigentlichen Textbereich.
- Man unterscheidet Kommandos der folgenden Kategorien:
 - Kommandos zur Formatierung des *kompletten Texts*.
 - Kommandos, die das aktuelle *Absatzformat* verändern.
 - Kommandos, die für die aktuelle *Textausgabe* gültig sind.

```
\paperw11906\paperh16838\margl1417\marginr1417\marginl1417  
\marginb1134 \deftab708\widowctr\ftnbj\enddoc\hyphhotz425  
\noxlattoyen\expshrt\noultr\spc\dntb\insbdb\nospaceforul
```

Text: RTF

2.9 Formatierungen für den kompletten Text

- Papierbreite in twips (1 twip = 1/20 Punkt = 1/1440 Zoll) (`\paperwN`, Standard 12240 twips)
- Papierhöhe in twips (`\paperhN`, Standard 15840 twips)
- Ränder (`\marglN`, `\margrN`, ..., Standard 1800 twips)
- Standardtabulatorweite (`\deftabN`, Standard 720 twips)
- ...

Twips, Inches und Zentimeter

- Twip (twentieth of a point) ist eine bildschirm-unabhängige Einheit über die die Proportionen und Positionen von Bildelementen auf allen Anzeigesystemen gleich ausgedrückt werden können.
- Entsprechungen:
 - 1 pica = 1/6 inch
 - 1 point = 1/12 pica
 - 1 twip = 1/20 point or 20 twips = 1 point
 - 1 twip = 1/567 centimeter or 567 twips = 1 centimeter
 - 1 twip = 1/1440 inch or 1440 twips = 1 inch
- Die Anzahl der twips pro pixel hängt von der verwendeten Hardware und Bildschirmauflösung ab (z.B. 800x600: ca. 15 twips/ pixel).

2.10 RTF-Abschnittsformate

- neuer Abschnitt (\sect)
- zurücksetzen der Standardeinstellungen des Abschnitts (\sectd)
- vertikale Position der Kopfzeile vom oberen Seitenrand (\headeryN)
- vertikale Position der Fußzeile vom unteren Seitenrand (\footeryN)

\sect\linex0\headery708\footery708\colsx708 ...

2.11 RTF-Absatzformate

- neuer Absatz (\par)
- Aufzählungen und Nummerierungen (\pn...)
- Fettdruck (\b)
- Kursivdruck (\i) ...

...

\viewkind4\uc1\pard\f0\fs20 DIES IST EIN WORDPAD DOKUMENT\par

\par

\b Fette Schrift\par

\b0i Kursive Schrift\par

\i0\f1 Anderer Schrifttyp\par

\f2\fs36 Gr'f6'dfere Schrift\par

\cf1\fs32 Farbe\f0\fs20\par

...

RTF-Beispiel

```
{\rtf1\ansi\ansicpg1252\uc1\deff0\stshfdbch0\stshfloch0\stshfhich0\stshfbi0\deflang1031\deflangfe1031
{\fonttbl {\f0\froman\fcharset0\prq<sub>2</sub> {\*\panose 02020603050405020304} Times New Roman;}
{\f34\fswiss\fcharset128\prq<sub>2</sub> {\*\panose 020b0604020202020204} Arial Unicode MS;}
{\f52\fswiss\fcharset128\prq<sub>2</sub> {\*\panose 020b0604020202020204} @Arial Unicode MS;}
{\f88\froman\fcharset238\prq<sub>2</sub> Times New Roman CE;}
{\f89\froman\fcharset204\prq<sub>2</sub> Times New Roman Cyr;}
{\f91\froman\fcharset161\prq<sub>2</sub> Times New Roman Greek;} ...
{\f430\fswiss\fcharset0\prq<sub>2</sub> Arial Unicode MS Western;}
{\f428\fswiss\fcharset238\prq<sub>2</sub> Arial Unicode MS CE;}
{\f429\fswiss\fcharset204\prq<sub>2</sub> Arial Unicode MS Cyr;}
{\f431\fswiss\fcharset161\prq<sub>2</sub> Arial Unicode MS Greek;} ...
{\f610\fswiss\fcharset0\prq<sub>2</sub> @Arial Unicode MS Western;}
{\f608\fswiss\fcharset238\prq<sub>2</sub> @Arial Unicode MS CE;}
{\f609\fswiss\fcharset204\prq<sub>2</sub> @Arial Unicode MS Cyr;}
{\f611\fswiss\fcharset161\prq<sub>2</sub> @Arial Unicode MS Greek;} ... } {\colortbl;\red0\green0\blue0;}
{\stylesheet } {\*\rsidtbl \rsid7417025\rsid7737050\rsid7941951\rsid11356226\rsid11417047\rsid12532369}
{\*\generator Microsoft Word 10.0.2627;} {\info {\title Dieses Dokument zeigt die Verwendung von RTF}
{\author Torsten Schassan} {\operator Torsten Schassan} {\creatim\yr2003\mo5\dy14\hr9\min31}
{\revtim\yr2003\mo5\dy14\hr9\min31} {\version2} {\edmins0} {\nofpages1} {\nofwords19} {\nofchars121}
{\nofcharsws139} {\vern16437} } \paperw11906\paperh16838\margl1417\margr1417\margt1417\margb1134
\deftab708\widowctrl\ftnbj\aeenddoc\hyphhotz425\noxlattoyen\expshrt\noultr\lspc\dntblnsbdb\nospaceforu\for
mshade\horzdoc\dgmargin\dghspace180\dgvspace180\dghorigin1417\dgvorigin1417\dghshow1\dgvshow1
\jexpand\viewkind1\viewscale117\viewzk2\pgbrdrhead\pgbrdrfoot\splytwnine\ftnlytwnine\htmautsp\nolnhtadjtbl
\useltbaln\alntblind\lytcalctblwd\lyttblrtgr\lnbrkrule\nobrkwrtbl\snaptogridincell\allowfieldendsel\wrppunct\asia
nbrkrule\rsidroot11417047 \fet0 \sectd \linex0\headery708\footery708\colsx708\endnhere\sectlinegrid360
\sectdefaultcl\ftnbj {\*\pnseclvl1\pnucrm\pnstart1\pnindent720\pnhang {\pntxta .}} ... \ql
\li0\ri0\widctlpar\aspalpha\aspnum\faauto\adjustright\rin0\lin0\itap0
\fs24\lang1031\langfe1031\cgrid\langnp1031\langfenp1031 {\insrsid7737050 Dieses Dokument zeigt die
Verwendung von RTF.}{\insrsid11356226 \par }{\insrsid7737050 \par Beispiel:
}{\b\insrsid7737050\charrsid7737050 Fettdruck}{\b\insrsid7737050 \par \par Beispiel:
}{\i\insrsid7737050\charrsid7737050 Kursiv}{\i\insrsid7737050 \par }{\insrsid7737050\charrsid7737050 \par
Beispiel: }{\loch\af34\hich\af34\dbch\af34\insrsid7737050\charrsid7737050 \hich\af34\dbch\af34\loch\af34
andere Schriftart}{\loch\af34\hich\af34\dbch\af34\insrsid7737050 \par \par \hich\af34\dbch\af34\loch\af34 Und:
}{\fs32\loch\af430\hich\af430\dbch\af34\insrsid7737050\charrsid7737050 \hich\af430\dbch\af34\loch\af430
\hich\af430 andere Schriftgr\`f6\df\loch\af430 e}{\insrsid7737050\charrsid7737050 \par }}
```

Text: PDF

3. PDF (Portable Document Format)

3.1 Charakteristik

- PDF ist ein proprietäres (Adobe Inc.) Dokumentenbeschreibungsformat das neben Text auch Grafiken u.a. Objekte auszeichnen kann.
- Die PDF-Formatdokumentation ist offen zugänglich; PDF benötigt zum Lesen ein Programm (Acrobat Reader).
- Das Format lehnt an PostScript an .
- Die Informationen sind in ASCII hinterlegt und eine Zeile darf nicht mehr als 255 Zeichen enthalten.
- Binäre Daten (z.B. für Bilder etc.) werden als (hexadezimale) ASCII-Zeichen kodiert.
- Eine PDF-Datei besteht aus
 - einem Header,
 - einem Body (dem Dokumenteninhalt),
 - einer Querverweisliste und
 - dem Trailer.
- PDF-Dateien sind nicht linear, d.h. der erste Absatz im Text muss nicht zwangsläufig auch am Anfang der Datei liegen.

Text: PDF

PDF-Beispieldatei

```
%PDF-1.2
3 0 obj
<< /Length 4 0 R /Filter /FlateDecode >>
stream ...
endobj 4 0 obj 1583 endobj 2 0 obj
<< /Type /Page /Contents 3 0 R /Resources 1 0 R /MediaBox [0 0
595.273 841.887] /Parent 12 0 R >>
endobj 1 0 obj
<< /Font << /F16 5 0 R /F17 6 0 R /F18 7 0 R /F19 8 0 R /F27 9
0 R /F42 10 0 R /F15 8 0 R /F43 11 0 R >> /ProcSet [ /PDF /Text
] >>
  endobj 15 0 obj
<< /Length 16 0 R /Filter /FlateDecode >>
stream ... trailer
<< /Size 475 /Root 473 0 R /Info 474 0 R >>
startxref 359455
%%EOF
```

Text: PDF

3.2 PDF-Befehle

3.2.1 Allgemein

- Alle Befehle und Namen sind *case-sensitiv*.
- Es können Werte (die als Objekte interpretiert werden) der Basistypen boolean, number, string, array, stream, u.a. verwendet werden.
- **Zahlen** werden als integer oder real dargestellt
- **Zeichenketten** werden in runde Klammern geschlossen oder über eine Sequenz aus Hexadezimalzahlen angegeben
 - (Dies ist ein Text)
 - <b64328e1c56ce225183bac5f0dff91bd>
- Der **Backslash** hat eine markierende Funktion: er leitet eine sog. **Escape-Sequenz** ein, das folgende Zeichen hat eine besondere Bedeutung.
 - \n = linefeed
 - \r = carriage return
 - \t = horizontaler Tabulator
 - \b = backspace
 - \\ = backslash
 - \ddd = alle mit ASCII nicht kodierbaren Zeichen;
PDF-Dateien enthalten nur ASCII-Zeichen, alle nicht darstellbaren Zeichen müssen durch \ddd dargestellt werden.

Text: PDF

3.2.2 PDF-Sequenzen und -Dictionaries

- PDF-Objekte können zu Sequenzen zusammengefaßt werden und in Arrays gespeichert werden.

→ [0 1 /test]

- Dictionaries sind Tabellen, in denen Objektpaare (key /value) zugeordnet werden.

Der erste Eintrag in ein Dictionary ist

/type

gefolgt vom Namen des Objektes. Es folgen Schlüssel-Werte-Paare.

- Namen werden in der Form

/name

angegeben.

→ << /type /name key1 value1 key2 value2 ...>>

Text: PDF

2.3 PDF-Streams

- Streams sind Zeichenfolgen, die durch die Anwendung sequentiell gelesen werden können.
- Streams eignen sich daher zur Speicherung großer Datenmengen, wie Bilder oder Seitenbeschreibungen.
- Ein Stream besteht aus einem Dictionary-Objekt, welches die Zeichenfolge beschreibt, gefolgt vom Schlüsselwort *stream*, gefolgt von dem eigentlichen Text und abgeschlossen vom Schlüsselwort *end-stream*.

→ <<dictionary object>> stream (Ein Text) endstream

Text

Aufgaben

- 1) **Wiederholen** Sie den Stoff dieser Sitzung **bis zur nächsten Sitzung** (siehe dazu den Link zur Sitzung auf der HKI-Homepage). Informieren Sie sich zusätzlich durch eigene Literaturrecherche!
- 2) Beantworten Sie die Fragen aus der Sammlung „**beispielhafte Klausurfragen**“ zur Rechnerkommunikation (soweit in dieser Sitzung behandelt).