

Extensible Markup Language

aus Wikipedia, der freien Enzyklopädie


Die **Extensible Markup Language** (engl. „erweiterbare Auszeichnungssprache“), abgekürzt **XML**, ist eine Auszeichnungssprache zur Darstellung hierarchisch strukturierter Daten in Form von Textdateien. XML wird u. a. für den plattform- und implementationsunabhängigen Austausch von Daten zwischen Computersystemen eingesetzt, insbesondere über das Internet.^[1]

Die vom World Wide Web Consortium (W3C) herausgegebene XML-Spezifikation (*Recommendation*, erste Ausgabe vom 10. Februar 1998, aktuell ist die fünfte Ausgabe vom 26. November 2008) definiert eine Metasprache, auf deren Basis durch strukturelle und inhaltliche Einschränkungen anwendungsspezifische Sprachen definiert werden. Diese Einschränkungen werden durch Schemasprachen wie DTD oder XML Schema ausgedrückt. Beispiele für XML-Sprachen sind: RSS, MathML, GraphML, XHTML, XAML, Scalable Vector Graphics (SVG), GPX, aber auch XML-Schema.

Ein XML-Dokument besteht aus Textzeichen, im einfachsten Fall in ASCII-Kodierung, und ist damit menschenlesbar. Binärdaten enthält es per Definition nicht.

Extensible Markup Language

```
<?xml version="1.0"?>
<quiz>
  <frage>
    Wer war der fünfte
    deutsche Bundespräsident?
  </frage>
  <antwort>
    Karl Carstens
  </antwort>
  <!-- Anmerkung: Wir
    brauchen mehr Fragen -->
</quiz>
```



Dateiendung: .xml

MIME-Type: application/xml, text/xml (deprecated)

Magische Zahl: 3C 3F 78 6D 6C hex
<?xml

Entwickelt von: World Wide Web Consortium

Art: Auszeichnungssprache

Erweitert von: SGML

Erweitert zu: XHTML, RSS, Atom

Website: 1.0 (Fifth Edition)
(<http://www.w3.org/TR/2008/REC-xml-20081126/>)
1.1 (Second Edition)
(<http://www.w3.org/TR/2006/REC-xml11-20060816/>)

Inhaltsverzeichnis

- 1 Fachbegriffe
 - 1.1 Element
 - 1.2 Wohlgeformtheit
 - 1.3 Gültigkeit (Validität)
 - 1.4 Parser
- 2 Aufbau eines XML-Dokuments
 - 2.1 Physischer Aufbau
 - 2.2 Logischer Aufbau
- 3 Klassifizierung von XML-Dokumenten
- 4 Verarbeitung von XML
 - 4.1 Verarbeitungskriterien

- 4.2 Programmgesteuerter Zugriff auf XML-Dokumente
- 4.3 XML-Parser-API-Beispiele
- 4.4 Transformation und Darstellung von XML-Dokumenten
- 5 Schemasprachen
 - 5.1 DTD
 - 5.2 XML Schema/XSD
 - 5.3 Weitere Schemasprachen
- 6 XML-Familie
 - 6.1 Infrastruktur
 - 6.2 Sprachen
 - 6.2.1 Text
 - 6.2.2 Grafik
 - 6.2.3 Geodaten
 - 6.2.4 Multimedia
 - 6.2.5 Sicherheit
 - 6.2.6 Ingenieurwissenschaften
 - 6.2.7 Weitere
- 7 Alternative Formate
- 8 Literatur
- 9 Weblinks
- 10 Einzelnachweise

Fachbegriffe

Element

Wichtigste Struktureinheit einer XML-Anwendung ist das *Element*. Der Name eines XML-Elements kann weitgehend frei gewählt werden. Elemente können weitere Elemente, Text- und andere Knoten – ggfs. auch vermischt – enthalten. Elemente sind die Träger der Information in einem XML-Dokument, unabhängig davon, ob es sich um Text, Bilder usw. handelt.

Wohlgeformtheit

Ein XML-Dokument heißt „wohlgeformt“ (oder englisch *well-formed*), wenn es alle XML-Regeln einhält. Beispielhaft seien hier folgende genannt:

- Das Dokument besitzt genau ein Wurzelement. Als Wurzelement wird dabei das jeweils äußerste Element bezeichnet, z. B. `<html>` in XHTML.
- Alle Elemente mit Inhalt besitzen einen Beginn- und einen End-Auszeichner (-Tag) (z. B. `<eintrag>Eintrag 1</eintrag>`). Elemente ohne Inhalt können auch in sich geschlossen sein, wenn sie aus nur einem Auszeichner bestehen, der mit `/>` abschließt (z. B. `<eintrag />`).
- Die Beginn- und End-Auszeichner sind ebenentreu-paarig verschachtelt. Das bedeutet, dass alle Elemente geschlossen werden müssen, bevor die End-Auszeichner des entsprechenden Elternelements oder die Beginn-Auszeichner eines Geschwisterelements erscheinen.
- Ein Element darf nicht mehrere Attribute mit demselben Namen besitzen.
- Attributeigenschaften müssen in Anführungszeichen stehen.
- Die Beginn- und End-Auszeichner beachten die Groß- und Kleinschreibung (z. B. `<eintrag>`

</Eintrag> ist nicht gültig).

Gültigkeit (Validität)

Soll XML für den Datenaustausch verwendet werden, ist es von Vorteil, wenn das Format mittels einer Grammatik (z. B. einer Dokumenttypdefinition oder eines XML-Schemas) definiert ist. Der Standard definiert ein XML-Dokument als gültig (oder englisch *valid*), wenn es wohlgeformt ist, den Verweis auf eine Grammatik enthält und das durch die Grammatik beschriebene Format einhält.

Parser

Programme oder Programmteile, die XML-Daten auslesen, interpretieren und ggf. auf Gültigkeit prüfen, nennt man *XML-Parser*. Prüft der Parser die Gültigkeit, so ist er ein *validierender* Parser.

Aufbau eines XML-Dokuments

Beispiel einer XML-Datei

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<verzeichnis>
  <titel>Wikipedia Städteverzeichnis</titel>
  <eintrag>
    <stichwort>Genf</stichwort>
    <eintragstext>Genf ist der Sitz von ...</eintragstext>
  </eintrag>
  <eintrag>
    <stichwort>Köln</stichwort>
    <eintragstext>Köln ist eine Stadt, die ...</eintragstext>
  </eintrag>
</verzeichnis>
```

XML-Dokumente besitzen einen physischen und einen logischen Aufbau.

Physischer Aufbau

- Entitäten. Die erste Entität ist die Hauptdatei des XML-Dokuments. Weitere mögliche Entitäten sind über
 - Entitätenreferenzen (*&name;* für das Dokument bzw. *%name;* für die Dokumenttypdefinition) eingebundene Zeichenketten, eventuell auch ganze Dateien, sowie Referenzen auf Zeichenentitäten zur Einbindung einzelner Zeichen, die über ihre Nummer referenziert wurden (*&#Dezimalzahl;* oder *&#xHexadezimalzahl;*).
- Eine XML-Deklaration wird optional verwendet, um XML-Version, Zeichenkodierung und Verarbeitbarkeit ohne Dokumenttypdefinition zu spezifizieren.
- Eine Dokumenttypdefinition wird optional verwendet, um Entitäten sowie den erlaubten logischen Aufbau zu spezifizieren.

Logischer Aufbau

Der logische Aufbau entspricht einer Baumstruktur und ist damit hierarchisch organisiert. Als Baumknoten gibt es:

- Elemente, deren physische Auszeichnung mittels

- einem passenden Paar aus Start-Tag (`<Tag-Name>`) und End-Tag (`</Tag-Name>`) oder
- einem Empty-Element-Tag (`<Tag-Name />`) erfolgen kann,
- Attribute als bei einem Start-Tag oder Empty-Element-Tag geschriebene Schlüsselwort-Werte-Paare (`Attribut-Name="Attribut-Wert"`) für Zusatz-Informationen über Elemente (eine Art Meta-Information),
- Verarbeitungsanweisungen (`<?Ziel-Name Parameter ?>`, engl. *Processing Instruction*),
- Kommentare (`<!-- Kommentar-Text -->`), und
- Text, der als normaler Text oder in Form eines CDATA-Abschnittes (`<![CDATA[beliebiger Text]]>`) auftreten kann.

Ein XML-Dokument muss genau ein Element auf der obersten Ebene enthalten. Unterhalb dieses Dokumentelements können weitere Elemente verschachtelt werden. Weiterhin kann durch Angabe eines *namespace* (XML-Namensraum) sichergestellt werden, dass bei Überschneidungen mit XML-Daten eines anderen Vokabulars keine Doppeldeutigkeiten entstehen.

Zur Spezifikation des logischen Aufbaus werden die Dokumenttypdefinitionen durch das umfangreichere XML-Schema abgelöst, welches keine Möglichkeit zur Definition von Entitäten, jedoch einen adäquaten Ersatz dafür besitzt. Verarbeitungsanweisungen werden in der Praxis meist eingesetzt, um in XML-Dokumenten Verarbeitungsanweisungen in anderen Sprachen einzubauen. Ein Beispiel dafür ist PHP, dessen Verarbeitungsanweisungen in XML-Dokumente mit einer PHP-Verarbeitungsanweisung, z. B. `<?php echo 'Hello, World'; ?>`, eingebaut werden können.

Einige Web-Browser, darunter Internet Explorer (MSXML engine), Mozilla Firefox und Netscape Navigator (TransforMiiX engine), Opera (native engine) und Safari, können XML-Dokumente mit Hilfe eines eingebauten XML-Parsers direkt darstellen. Dies geschieht z. B. in Verbindung mit einem Stylesheet. Diese Transformation kann die Daten in ein komplett anderes Format umwandeln, das Zielformat muss nicht einmal XML sein.

Klassifizierung von XML-Dokumenten

XML-Dokumente lassen sich anhand ihres beabsichtigten Gebrauchs und ihres Strukturierungsgrads in dokumentenzentrierte und datenzentrierte Dokumente unterteilen. Die Grenze zwischen diesen Dokumentenarten ist jedoch fließend. Mischformen können als *semistrukturiert* bezeichnet werden.

- dokumentenzentriert: Das Dokument ist an ein Textdokument angelehnt, das für den menschlichen Leser größtenteils auch ohne die zusätzliche Metainformation verständlich ist. XML-Elemente werden hauptsächlich zur semantischen Markierung von Passagen des Dokuments genutzt, das Dokument ist nur schwach strukturiert. Aufgrund der schwachen Strukturierung ist eine maschinelle Verarbeitung schwierig.
- datenzentriert: Das Dokument ist hauptsächlich für die maschinelle Verarbeitung bestimmt. Es folgt einem Schema, das Entitäten eines Datenmodells beschreibt und definiert, in welcher Beziehung die Entitäten zueinander stehen, sowie, welche Attribute die Entitäten haben. Das Dokument ist somit stark strukturiert und für den unmittelbaren menschlichen Gebrauch weniger geeignet.
- semistrukturiert: Semistrukturierte Dokumente stellen eine Art Mischform dar, die stärker strukturiert ist als dokumentenzentrierte Dokumente, aber schwächer als datenzentrierte Dokumente.

Es ist typisch für datenzentrierte XML-Dokumente, dass Elemente entweder Elementinhalt oder Textinhalt haben. Der sogenannte gemischte Inhalt (*mixed content*), bei dem Elemente sowohl Text als auch Kindelemente enthalten, ist für die anderen XML-Dokumente typisch.

Verarbeitung von XML

Verarbeitungskriterien

Grundsätzlich sind drei Aspekte beim Zugriff auf ein XML-Dokument von Bedeutung:

- Wie erfolgt der Zugriff auf die XML-Datei: sequenziell oder wahlfrei?
- Wie ist der Ablauf beim Zugriff auf die XML-Daten gestaltet: „Push“ oder „Pull“? (Push bedeutet, dass die Ablaufkontrolle des Programms beim Parser liegt. Pull bedeutet, dass die Ablaufkontrolle im Code, der den Parser aufruft, implementiert ist.)
- Wie erfolgt das Baumstrukturmanagement der XML-Daten: hierarchisch oder verschachtelt?

Programmgesteuerter Zugriff auf XML-Dokumente

Das Einlesen von XML-Dokumenten erfolgt auf unterster Ebene über eine spezielle Programmkomponente, einen XML-Prozessor, auch XML-Parser genannt. Er stellt ein API zur Verfügung, über das die Anwendung auf das XML-Dokument zugreift.

Die XML-Prozessoren unterstützen dabei drei grundlegende Verarbeitungsmodelle.

- DOM: Ein DOM-API repräsentiert ein XML-Dokument als Baumstruktur und gewährt wahlfreien Zugriff auf die einzelnen Bestandteile der Baumstruktur. DOM erlaubt außer dem Lesen von XML-Dokumenten auch die Manipulation der Baumstruktur und das Zurückschreiben der Baumstruktur in ein XML-Dokument. Aus diesem Grund ist DOM sehr speicherintensiv.
- SAX: Ein SAX-API repräsentiert ein XML-Dokument als sequentiellen Datenstrom und ruft für im Standard definierte Ereignisse vorgegebene Rückruffunktionen (callback function) auf. Eine Anwendung, die SAX nutzt, kann eigene Unterprogramme als Rückruffunktionen registrieren und auf diese Weise die XML-Daten auswerten.
- Pull-API: Ein XML-Pull-API verarbeitet Daten sequenziell und bietet sowohl ereignisbasierte Verarbeitung als auch einen Iterator an. Es ist hoch speichereffizient und ggf. leichter zu programmieren als das SAX-API, da die Ablaufkontrolle beim Programm und nicht beim Parser liegt.

Weitere Verarbeitungsmodelle:

- Data Binding: Diese Möglichkeit stellt XML-Daten als Datenstruktur direkt für einen Programmmzugriff bereit. Die XML-Daten werden per Unmarshalling direkt in z. B. Objekte gewandelt.
- Nicht extrahierendes XML-API: Die Daten werden auf Byte-Ebene sehr effizient verarbeitet.

Oftmals greift der Anwendungscode nicht direkt auf die Parser-API zu. Stattdessen wird XML weiter gekapselt, so dass der Anwendungscode mit nativen Objekten / Datenstrukturen arbeitet, welche sich auf XML abstützen. Beispiele für solche Zugriffsschichten sind JAXB in Java, der Data Binding Wizard in Delphi oder das XML Schema Definition Toolkit in .Net. Die Umwandlung von Objekten in XML ist üblicherweise bidirektional möglich. Diese Umwandlung wird als Serialisierung oder Marshalling bezeichnet.

XML-Parser-API-Beispiele

XML-Parser-APIs sind für verschiedene Programmiersprachen vorhanden, z. B. Java, C, C++, C#, Python, Perl und PHP. Parser-API-Beispiele:

- XML::Parser (Perl): Ein XML-Parser für Perl. Ein sehr einfaches API bietet z. B. auch das CPAN-Modul XML::Simple an.
- DOM Functions (PHP5): Modul in PHP5, um XML-Dokumente einzulesen; alternativ simpleXML; für

PHP4 gibt es DOM XML.

- StAX (Java): Eine hochgradig speichereffiziente Parserimplementierung (Pull) und gleichzeitig einfach zu programmieren. Es werden Cursor- und Iteratorverarbeitungsmodelle angeboten.
- JAXB: Data Binding für Java. Beispielsweise kann aus einem XML-Schema die entsprechende Java-Klasse generiert werden und umgekehrt.
- Apache XMLBeans Java Data Binding Framework, kann bereits mit Java 1.4.2 verwendet werden
- Xerces: Ein validierender XML-Parser für C++, Java und Perl für eine große Anzahl an Plattformen.
- ElementTree iterparse (<http://effbot.org/zone/element-iterparse.htm>): Ein Parser-API für Python, die über Teilbäume iteriert. Es kombiniert die Speichereffizienz eines Pull-Parsers mit der Einfachheit eines DOM-Parsers.
- VTD-XML: Beispiel für ein nicht extrahierendes XML-API.
- MSXML: Microsoft XML Core Services, die Microsoft XML Softwarebibliothek für XML-Unterstützung über DOM, SAX, XSLT, XML Schemata und andere zu XML gehörende Technologien

Zur Erstellung von XML-Dokumenten gibt es spezielle Programme, sogenannte XML-Editoren. Zur Speicherung und Verwaltung von XML-Dokumenten gibt es ebenfalls spezielle Programme, sogenannte XML-Datenbanken.

Transformation und Darstellung von XML-Dokumenten

Ein XML-Dokument kann mittels geeigneter Transformationssprachen wie XSLT oder DSSSL in ein anderes Dokument transformiert werden. Oftmals dient die Transformation zur Überführung eines Dokuments aus einer XML-Sprache in eine andere XML-Sprache, beispielsweise zur Transformation nach XHTML, um das Dokument in einem Webbrowser anzuzeigen.

Schemasprachen

Um die Struktur von XML-Sprachen zu beschreiben, bedient man sich so genannter *Schemasprachen*. Die zwei bekanntesten sind Dokumenttypdefinition und XML Schema.

DTD

→ *Hauptartikel: Dokumenttypdefinition*

Eine Dokumenttypdefinition (DTD) beschreibt die Struktur und Grammatik von XML-Dokumenten. Sie wurde zusammen mit XML standardisiert, zu einem Zeitpunkt, an dem XML noch hauptsächlich für „*narrative documents*“ („erzählende Dokumente“, also Zeitungsartikel, Bücher, ...) gedacht war, weniger als Datenaustauschformat. Daher ist es z. B. in DTD nicht möglich, zwischen Texten und Zahlen zu unterscheiden. Ein weiterer Nachteil ist die Tatsache, dass die DTD in einer eigenen Sprache abgefasst werden muss. Außerdem kennt die DTD keine Namensräume.

XML Schema/XSD

→ *Hauptartikel: XML Schema*

XML Schema (beziehungsweise XSD für XML-Schema-Definition) ist die moderne Möglichkeit, die Struktur von XML-Dokumenten zu beschreiben. XML Schema bietet auch die Möglichkeit, den Inhalt von Elementen und Attributen zu beschränken, z. B. auf Zahlen, Datumsangaben oder Texte, z. B. mittels regulärer Ausdrücke.

Ein Schema ist selbst ein XML-Dokument, welches erlaubt, komplexere (auch inhaltliche) Zusammenhänge zu beschreiben, als dies mit einer formalen DTD möglich ist.

Weitere Schemasprachen

Weitere Schemasprachen sind Document Structure Description, RELAX NG und Schematron.

XML-Familie

Infrastruktur

Im Zusammenhang mit XML wurden vom W3-Konsortium auf Basis von XML viele Sprachen definiert, welche XML-Ausdrücke für häufig benötigte allgemeine Funktionen anbieten wie etwa die Verknüpfung von XML-Dokumenten. Zahlreiche XML-Sprachen nutzen diese Grundbausteine.

- Transformation von XML-Dokumenten: XSLT, STX
- Adressierung von Teilen eines XML-Baumes: XPath
- Verknüpfung von XML-Ressourcen: XPointer, XLink und XInclude
- Selektion von Daten aus einem XML-Datensatz: XQuery
- Datenmanipulation in einem XML-Datensatz: XUpdate
- Abfassen von elektronischen Formularen: XForms
- Definition von XML-Datenstrukturen: XML Schema (= XSD, XML Schema Definition Language), DTD und RELAX NG
- Signatur und Verschlüsselung von XML-Knoten: XML Signature und XML-Encryption
- Aussagen zum formellen Informationsgehalt: XML Infoset
- Formatierte Darstellung von XML-Daten: XSL-FO
- Definition zum Methoden- bzw. Funktionsaufruf durch verteilte Systeme: XML-RPC
- Standardisierte Attribute: XML Base und ID (DTD)
- XML-basierte deklarative Programmiersprache: MXML

Sprachen

Während XML selbst aus SGML hervorgegangen ist, bedienen sich heute sehr viele formale Sprachen der Syntax von XML. So ist XML ein wesentliches Instrument, um – wie es das W3C vorsieht – eine offene, für Mensch und Maschine verständliche Informationslandschaft (semantisches Web) zu schaffen.

Auch die bekannte Dokumentsprache HTML wurde als „Extensible HyperText Markup Language“ (XHTML) im Anschluss an die Version 4.01 in dieses Konzept integriert, so dass ihr nun XML als Definitionsbasis zu Grunde liegt. Vielfacher Grund für den Einsatz von XML ist das zahlreiche Vorhandensein von Parsern und die einfache Syntax: die Definition von SGML umfasst 500 Seiten, jene von XML nur 26.

Die folgenden Listen stellen einige dieser XML-Sprachen dar.

Text

- XSL-FO (Textformatierung)
- DocBook
- DITA

- XHTML (XML-konformes HTML)
- TEI (Text Encoding Initiative)
- NITF (News Industry Text Format)
- OPML (Outline Processor Markup Language)
- OSIS (Open Scripture Information Standard)

Grafik

- SVG (Vektorgrafiken)
- X3D (3D-Modellierungssprache)
- Collada (Austauschformat für Daten zwischen verschiedenen 3D-Programmen)

Geodaten

- Geography Markup Language (GML)
- GPS Exchange Format (GPX): XML für GPS-Daten
- Keyhole Markup Language (KML): Koordinaten-Spezifikation für Google Earth
- City Geography Markup Language (CityGML)
- OpenStreetMap (OSM)

Multimedia

- MusicXML (Notendaten, aufgeschriebene Musik)
- SMIL (zeitsynchronisierte, multimediale Inhalte)
- MPEG-7 (MPEG-7 Metadaten)
- Laszlo (LZX)

Sicherheit

- Security Assertion Markup Language (sicherheitsbezogene Informationen beschreiben und übertragen)
- XML Signature (XML-Schreibweise für digitale Signaturen)
- XML Encryption

Ingenieurwissenschaften

- AutomationML, ein Format zur Speicherung von Anlagenplanungsdaten
- CAEX, ein Format zur Speicherung hierarchischer Objektinformationen
- GSDML, ein Format zur Beschreibung von Automatisierungsgeräten, die mit Profinet kommunizieren können
- IODD, ein Format zur Beschreibung von Sensoren und Aktoren

Weitere

Darüber hinaus existieren XML-Sprachen für Webservices (z. B. SOAP, WSDL und WS-*), für die Einbindung von Java-Code in XML-Dokumente (XSP), für die Synchronisation von Kalenderdaten SyncML, mathematische Formeln (MathML), Repräsentation von Graphen (GraphML), Verfahren im Bereich des Semantischen Webs (RDF, OWL, Topic Maps, UOML), *Service Provisioning* (SPML), den Austausch von Nachrichten (XMPP) oder Finanzberichten wie bspw. Jahresabschlüssen (XBRL), in Bereichen der

Automobilindustrie (ODX, MSRSW, AUTOSAR-Templates, QDX, JADM, OTX), automatisierter Test z. B. von Schaltkreisen (ATML) über Systembiologie (SBML) sowie Landwirtschaft (AgroXML) bis zum Verlagswesen (ONIX) oder Chemie (CIDX) und viele weitere mehr.

Eine Zusammenfassung von XML-Sprachen für Office-Anwendungen findet sich im OpenDocument-Austauschformat (*OASIS Open Document Format for Office Applications*).

Alternative Formate

- JSON (JavaScript Object Notation)
- YAML (YAML Ain't Markup Language)

Literatur

- Elliotte Rusty Harold: *Die XML Bibel*. 2. Auflage. mitp, 2002, ISBN 978-3-8266-0821-6.
- Stefan Mintert: *XML & Co*. Die W3C-Spezifikationen für Dokumenten- und Datenarchitektur. Addison-Wesley, München, ISBN 3827318440.
- Christine Kränzler: *XML/XSL - ... für professionelle Einsteiger . für Buch und Web*. Markt+Technik, München/Germany 2002, ISBN 978-3-8272-6339-1.
- Henning Lobin: *Informationsmodellierung in XML und SGML*. Springer, Berlin/Germany 2000, ISBN 3-540-65356-2.
- Erik T. Ray: *Einführung in XML*. O'Reilly, 2004, ISBN 3-8972-1286-2.
- Helmut Vonhoegen: *Einstieg in XML*. Aktuelle Standards: XML Schema, XSL, XLink. 7. Auflage. Galileo Press, 2013, ISBN 978-3-8362-2620-2.
- Frank Bitzer: *XML im Unternehmen*. Briefing fürs IT-Management. Galileo Press, Bonn 2002, ISBN 978-3-8984-2288-8.
- Michael Seeboerger-Weichselbaum: *Das Einsteigerseminar XML*. 2. Auflage. BHV Software, Kaarst 2000, ISBN 978-3-8287-1018-4.
- Margit Becher: *XML : DTD, XML-Schema, XPath, XQuery, XSLT, XSL-FO, SAX, DOM*. W3L Verlag, Witten 2009, ISBN 978-3-937137-69-8.
- Marco Skulchus, Marcus Wiederstein: *XML: Standards und Technologien*. Comelio Medien, Berlin 2009, ISBN 978-3-939701-21-7.

Weblinks

 **Commons: XML** ([//commons.wikimedia.org/wiki/Category:XML?uselang=de](http://commons.wikimedia.org/wiki/Category:XML?uselang=de)) – Sammlung von Bildern, Videos und Audiodateien

 **Wikibooks: Websiteentwicklung: XML** – Lern- und Lehrmaterialien

- World Wide Web Consortium über XML (<http://www.w3.org/XML>) (englisch), edition-w3c.de – Deutsche Übersetzungen zu XML u. a. (<http://www.edition-w3c.de/>)
- Links zum Thema XML (http://www.dmoz.org/World/Deutsch/Computer/Datenformate/Markup_Languages/XML/) im Open Directory Project
- Infos zu diversen XML-Technologien wie XSLT, XPath, Schematron, XProc, WordML, XSL-FO (<http://www.data2type.de/xml-xslt-xslfo/>)

Einzelnachweise

1. Tim Bray: *Extensible Markup Language (XML) 1.0 (Fourth Edition) – Origin and Goals*. (<http://www.w3.org/TR/2006/REC-xml-20060816/#sec-origin-goals>) World Wide Web Consortium, September 2006, abgerufen am 9. April 2013.

Normdaten (Sachbegriff): GND: 4501553-3

Von „http://de.wikipedia.org/w/index.php?title=Extensible_Markup_Language&oldid=121998768“

Kategorien: XML | Standard für Elektronischen Datenaustausch | Beschreibungssprache

- Diese Seite wurde zuletzt am 28. August 2013 um 15:54 Uhr geändert.
- Abrufstatistik

Der Text ist unter der Lizenz „Creative Commons Attribution/Share Alike“ verfügbar; Informationen zum Lizenzstatus eingebundener Mediendateien (etwa Bilder oder Videos) können im Regelfall durch Anklicken dieser abgerufen werden. Möglicherweise unterliegen die Inhalte jeweils zusätzlichen Bedingungen. Durch die Nutzung dieser Website erklären Sie sich mit den Nutzungsbedingungen und der Datenschutzrichtlinie einverstanden.

Wikipedia® ist eine eingetragene Marke der Wikimedia Foundation Inc.