EXist

Wechseln zu:Navigation, Suche

Inhaltsverzeichnis

1 Definition

Das Open-Source Projekt **eXist** bietet ein in Java implementiertes Datenbank-Management-System (DBMS), welches seine Daten nativ im XML-Datenmodell abspeichert und verwaltet. Es stützt sich ausschließlich auf freie W3C-Standards wie XML Query Language und XML Path Language als Abfragesprachen oder XSLT als Transformationssprache.

Begründet wurde *eXist* 2000 durch Wolfgang Meier, der dessen Entwicklung seit 2001 auf dem Open-Source-Portal SourceForge.net gemeinschaftlich vorantreibt. Es steht unter der Lizenzierung der GNU Lesser General Public License (LGPL) und ist in diesem Rahmen also auch in kommerziellen Projekten als Datenbank-Backend frei nutzbar.

2 Leistungsumfang

Die eXist XML-Datenbank implementiert eine breite Palette an offenen Webstandards, um ihre Leistungen zu erreichen:

Persitieren: Generell textuelle und multimediale Inhalte sowie optimiert XML-Daten
Selektion: XQuery 1.0 oder XPath 2.0
Modifikation: XUpdate und bald die XQuery Update Facility 1.0
Transformation: XSLT 1.0 oder XSLT 2.0
HTTP-Schnittstellen: REST, WebDAV, SOAP, XMLRPC, Atom Publishing
Erweiterungen: XQuery Funktionsbibliotheken als Erweiterungsmodule

2.1 REST-Schnittstelle

In Form eines Java-Servlets realisiert eXist eine Zugriffsschnittstelle, die sich am *Representational State Transfer (REST)* als Architekturstil für Hypermediasysteme anlehnt. Es veröffentlicht darüber den Verzeichnisbaum ihres Speicherbereichs unter einer Basis-URI. Befinden sich Anwendungsdateien z.B. unter dem Datenbankpfad `/cms/`, so werden sie bei eXist standardmäßig über `http://host:port/exist/rest/db/cms/` netzwerkweit mit einem HTTP-Client zugreifbar -

selbstverständlich mit optionaler HTTP-Authentifizierung. Über die einzelnen HTTP-Methoden lassen sich auf solchen Datenbank-URIs verschiedene Operationen auslösen:

GET: Bezieht je nach HTTP-Header *Content-Type* eine passende Repräsentation des Datenbankobjekts (z.B. ”text/xml”). Übergibt man einen XQuery-Ausdruck als Request-Parameter *_query*, so wird stattdessen dieser ausgeführt und sein Resultat zurückgegeben.
POST: Kann im POST-Inhalt eine für die GET-Variante zu lange XQuery enthalten, die in spezielle *<exist:query>*-Tags einzubetten ist und äquivalent zu *_query* behandelt wird.
PUT: Speichert oder ersetzt ein Dokument als Datenbankobjekt, wobei der HTTP-Header *Content-Type* über das gewählte textuelle oder binäre Ablageformat entscheidet.
DELETE: Entfernt das addressierte Objekt aus der Datenbank, falls existent.

Ein Spezialverhalten weisen binär abgelegte XQuery-Dokumente (*.xql) auf, bei denen GET und POST direkt in der Auswertung ihres Inhalts resultieren. Dies erlaubt es über XQuery zusammengestellte Webseiten direkt auszuliefern oder z.B. Formularanfragen aus XForms zu bedienen.

2.2 XQuery-Erweiterungen

Für die Umsetzung von Websystemen, die z.B. über die REST-Schnittstelle abgespeicherte XQuerys abfragen, bietet eXist einige nützliche Erweiterungen als XQuery-Funktionsbibliotheken. Neben recht speziellen Leistungen wie Emailversand, Datenkompression, Bilderverwaltung und mathematischen Funktionen sorgen HTTP-Kontextzugriff und Utility-Funktionen für den höchsten Mehrwert.

HTTP-Kontextzugriff: Erlaubt das Auslesen und Manipulieren der im aktuellen HTTP-Kontext befindlichen Informationen.

Request/Response: Setter & Getter für Header, Parameter, Attribute, Daten, Pfadinformationen, Statuscode und Cookies.
Sessions: Lebenszyklusverwaltung sowie Setter & Getter für Attribute.
Sonstiges: URL-Encoding & -Decoding, Redirects sowie Streaming von Binärdaten.

Utility-Funktionen: Nützliche Funktionen, die hauptsächlich die dynamische Datengenerierung, effiziente Datenverwaltung und funktionale Programmierung anreichern.

Datenverwaltung: Einlesen von textuellen und binären Dokumenten, Umwandeln zwischen textuellen und binären Daten, Parsen zu logischem XML-Baum, Serialisierung einer Knotensequenz.
Dynamik & Programmierung: Dynamische Modulimporte und XQuery-Auswertung, Locking für Knoten, Logging, MD5-Hashing, Zufallsgenerator, Funktionszeiger, Catch für Exceptions.

3 Quellen

Cagle, Kurt (2006): I Think, Therefore I eXist ... . O’Reilly Media Inc., XML.com, Onlineartikel.
Fielding, Roy T. (2000): Architectural Styles and the Design of Network-based Software Architectures. University of California, Irvine, Onlineversion.
Meier, Wolfgang (2003): eXist: An Open Source Native XML Database. Darmstadt University of Technology, Onlineversion.
SourceForge.net - eXist Open Source Native XML Database

Kategorien:
XML
Web-Programmierung

Content-Management
Glossar