

XML and databases (and XForms)

Patryk Czarnik

XML and Applications 2013/2014
Week 13 - 13.01.2014

XML support in databases - categorisation

- Classic (usually relational) database with XML support
 - logical structure - relations and references
 - additional XML-related features
 - used for application integration or storing XML data as part of larger data structures
- `Native' XML database
 - logical structure - collection of XML document trees
 - XQuery (or XPath) as native query language
 - natural XML-related features
 - used for storing XML data (or structural data easily mapped to XML tree)

XML support in relational databases

- Possible functionalities
 - import and export of data in XML format
 - special treatment of XML data stored in fields
 - XML validation as part of integrity constraints checking
 - XPath or XQuery for querying field contents
 - XSLT applied to query result
- How to store XML data?
 - whole document (fragment) stored in single field
 - split into prima factors
 - each XML node in separate field
 - tables structure reflects tree structure of XML

Example - XML support in Oracle database

- Since Oracle 8i (<http://www.oracle.com/xml>)
 - details differ from version to version
- XML parsers
 - for database programming (PL/SQL)
 - or middleware programming (Java, C++)
- XML-SQL Utility
 - XML data import and export
- **XMLType** data type and XML-specific operations

XML-SQL Utility

- getXML () function - XML data export

```
SELECT xmlgen.getXML('select * from emp') FROM dual;
```

```
<rowset>
  <row id="1">
    <empno>10</empno>
    <name>Scott Tiger</name>
    <title>specialist</title>
  </row>
  ...
</rowset>
```

XML in Oracle DB - XMLType

- **XMLType** – special datatype:
 - to be stored as LOB or used for columns, variables, etc.
 - indexing XML content
 - XPath expressions
 - validation against XML Schema
 - XSLT
- Available functions:
 - `extract`, `extractValue`, `existsNode`, `transform`, `updateXML`, `XMLSequence`

XMLType applications - some examples

```
CREATE TABLE warehouses(  
  warehouse_id NUMBER(4),  
  warehouse_spec XMLTYPE,  
  warehouse_name VARCHAR2(35),  
  location_id NUMBER(4));
```

```
CREATE TABLE po_xtab OF XMLType;
```

```
UPDATE po_xml_tab  
SET poDoc = UPDATEXML(poDoc,  
  '/PO/CUSTNAME/text()', 'John');
```

```
INSERT INTO warehouses VALUES  
(100, XMLType(  
  '<Warehouse whNo="100">  
    <Building>Owned</Building>  
  </Warehouse>'), 'Tower Records', 1003);
```

```
SELECT e.poDoc.getClobval() AS poXML  
FROM po_xml_tab e  
WHERE e.poDoc.existsNode('/PO[PNAME = "po_2"]') = 1;
```

```
CREATE INDEX city_index ON po_xml_tab  
(poDoc.extract('//PONO/text()').getNumberVal());
```

XML support in database engines

- Specification: ISO/IEC 9075-14:2011 *SQL/XML*
new data type **XML**:
 - only well-formed XML documents allowed
 - parsing and serialisation
 - implementation may add XML-specific operations
- Substantial support
 - IBM DB2 (since v.9 – *pureXML*)
 - Oracle (since 8i)
 - Microsoft SQL Server (since v.2000)
 - Sybase ASE (since v.12)
- Minimal support
 - MySQL – XPath queries over text fields containing XML
 - PostgreSQL – as above plus **XML** datatype but with no special operations

Native XML database

- Logical layer
 - XML document as basic data entity
 - collections of documents build a database
 - XML schema (or equivalent) as structure definition
 - XQuery (or XPath) as “native” query language
- Physical layer – not necessarily files with XML text
- More than just a collection XML files:
 - transactions and concurrent access
 - security (access privileges etc.), versioning, replication, ...
 - API for data access and update
 - additional means of data access
 - e.g. REST-compliant HTTP server
 - indexing for efficient access to selected nodes

Standards for XML databases

- High level query languages:
 - XQuery – primary language for queries
 - versions 1.0 and 3.0 in use
 - XQL – former approach to make XML query language
 - XPath – poor stub for XQuery
- High level update languages:
 - XQuery Update Extension
 - XUpdate
- Programmer APIs
depend additionally on programming language
 - XML Database API (XAPI)
 - XQJ (for Java, expected to become XML equivalent of JDBC)
 - vendor-specific APIs...

XML:DB

- Initiative for XML database interfaces specification
 - XML Database API (XAPI)
 - accessing XML databases from programs
 - resource collections (resource = XML document)
 - reading and writing documents via DOM or SAX
 - pluggable “services”; specified: XPath, transactions, operations on collections
 - last version: 2001
- XML Update Language (XUpdate)
 - XML application for updating XML databases
 - inserting, updating and removing nodes
 - XPath used for node addressing
 - last version: 2000

XUpdate - example

- Example (from XUpdate documentation)

```
<?xml version="1.0"?>
<xupdate:modifications version="1.0"
  xmlns:xupdate="http://www.xmldb.org/xupdate">
  <xupdate:insert-after select="/addresses/address[1]" >
    <xupdate:element name="address">
      <xupdate:attribute name="id">2</xupdate:attribute>
      <fullname>Lars Martin</fullname>
      <born day='2' month='12' year='1974' />
      <town>Leipzig</town>
      <country>Germany</country>
    </xupdate:element>
  </xupdate:insert-after>
  ...
</xupdate:modifications>
```

XML database products – overview

Product	Licence	Queries	XML:DB API
eXist	open source	XPath, XQuery	yes
BaseX	open source	XPath, XQuery	yes
MarkLogic	commercial		
Apache XIndex	open source	XPath	yes
Sedna	open source	XPath, XQuery	yes
Gemfire Enterprise	commercial	XQuery, OQL	yes
Tamino	commercial	XQuery, XPath	part

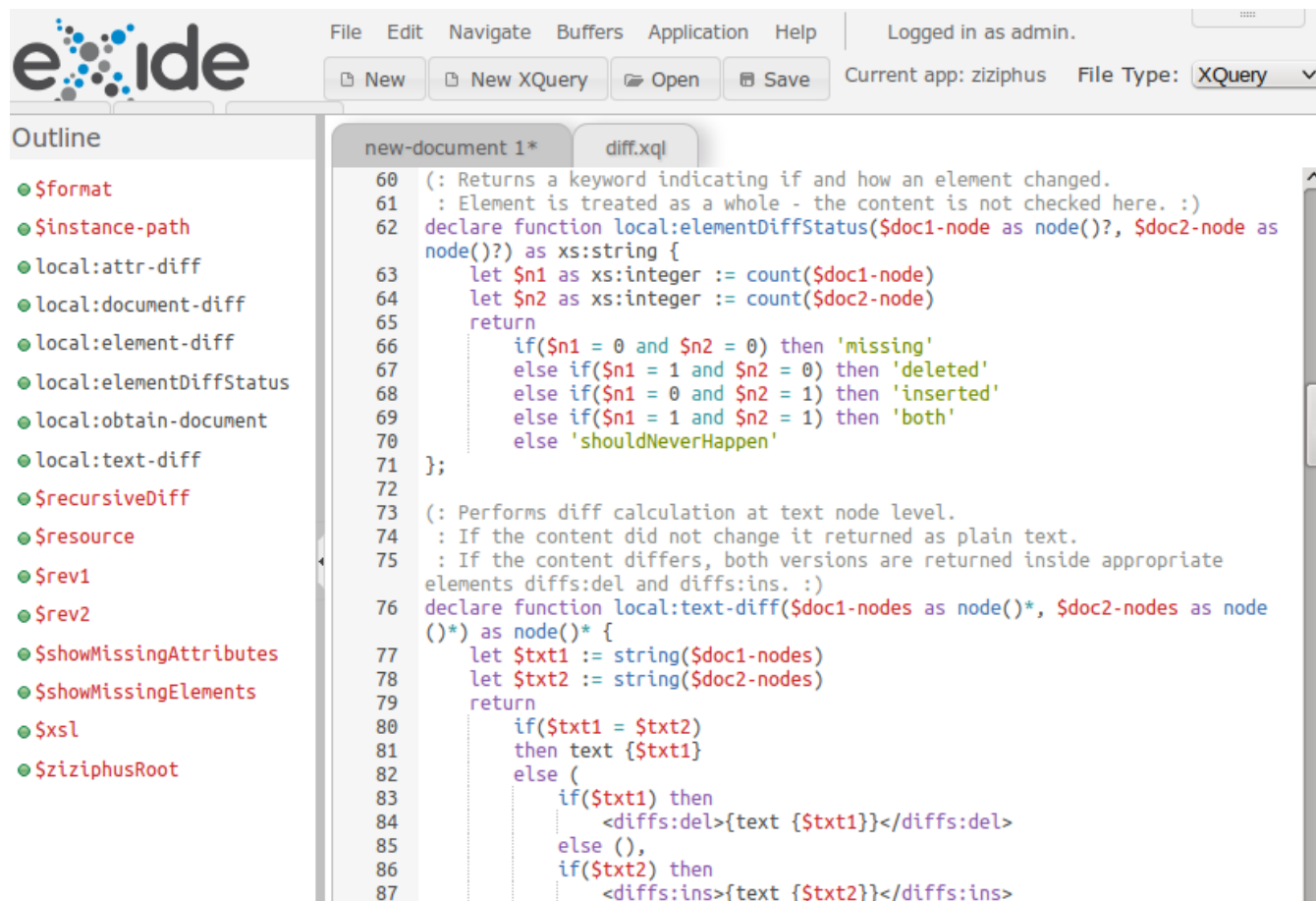
Source: Wikipedia and providers' websites

eXist DB

- One the most popular and the most elaborated XML database engines
- Open-source, but developed and supported by a (German) company
- Features include:
 - storage of XML and binary entities
 - various means of access, including: human-readable Web interface, direct HTTP access (REST-compliant), SOAP and XML-RPC, Java API (XQJ, elements of XAPI)
 - full XML model available in XPath, XQuery, and XSLT code
 - full XQuery support with majority of new 3.0 features, Update extension and some other non-standard extensions
 - XForms support using betterFORM or XSLTForms plugins

eXist - eXide

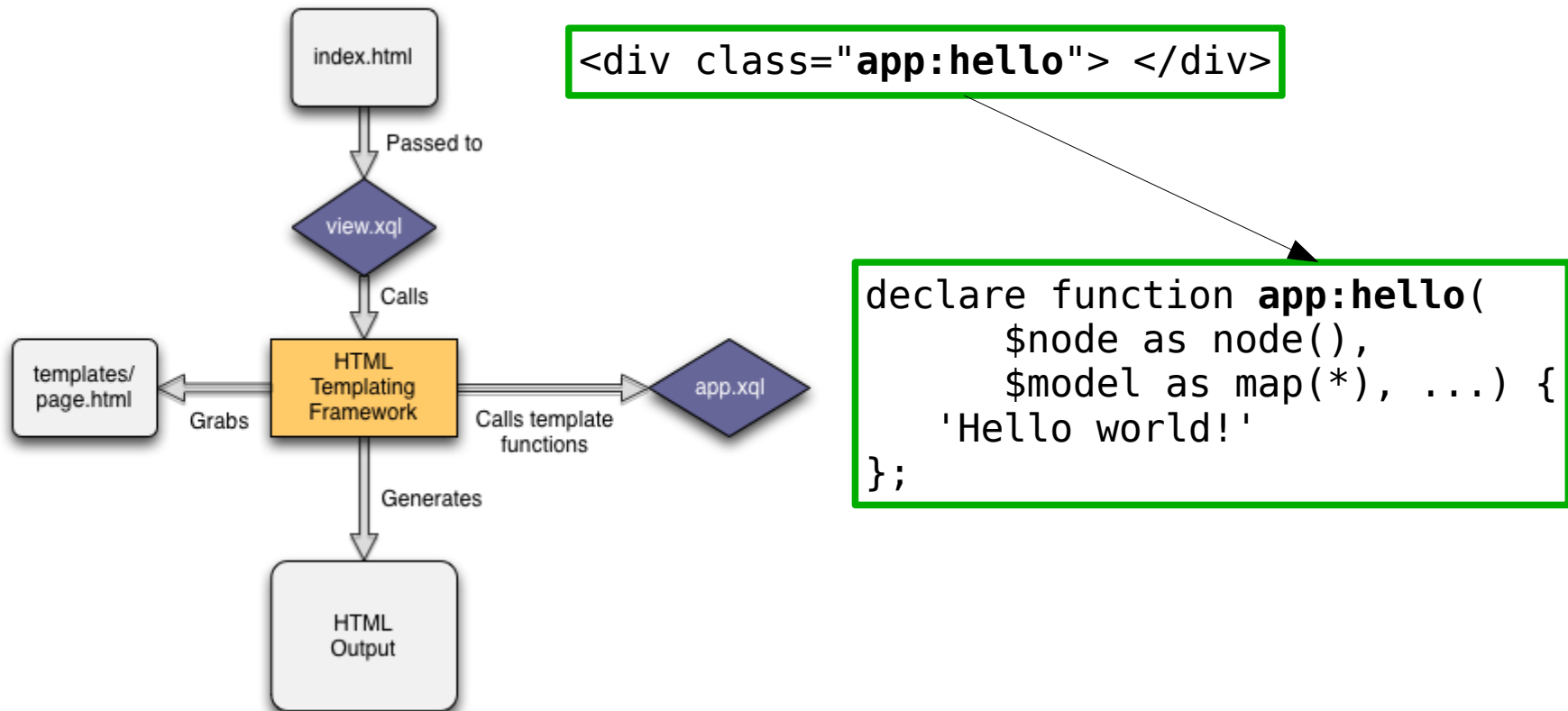
- XQuery programmer SDK running within a browser
 - supports also (to some extent...) XSLT, XML Schema, XHTML, XForms



```
60 (: Returns a keyword indicating if and how an element changed.
61 : Element is treated as a whole - the content is not checked here. :)
62 declare function local:elementDiffStatus($doc1-node as node()?, $doc2-node as
node())? as xs:string {
63   let $n1 as xs:integer := count($doc1-node)
64   let $n2 as xs:integer := count($doc2-node)
65   return
66     if($n1 = 0 and $n2 = 0) then 'missing'
67     else if($n1 = 1 and $n2 = 0) then 'deleted'
68     else if($n1 = 0 and $n2 = 1) then 'inserted'
69     else if($n1 = 1 and $n2 = 1) then 'both'
70     else 'shouldNeverHappen'
71 };
72
73 (: Performs diff calculation at text node level.
74 : If the content did not change it returned as plain text.
75 : If the content differs, both versions are returned inside appropriate
elements diffs:del and diffs:ins. :)
76 declare function local:text-diff($doc1-nodes as node()* , $doc2-nodes as node
()* ) as node()* {
77   let $txt1 := string($doc1-nodes)
78   let $txt2 := string($doc2-nodes)
79   return
80     if($txt1 = $txt2)
81     then text {$txt1}
82     else (
83       if($txt1) then
84         <diffs:del>{text {$txt1}}</diffs:del>
85       else (),
86       if($txt2) then
87         <diffs:ins>{text {$txt2}}</diffs:ins>
```

eXist - template mechanism

- Easy integration of XQuery logic and HTML interface



REST services - recall

- REST for Representational State Transfer
- Principles:
 - Service = collection of resources
 - URL identifies a resource
 - Resource has a normalised representation and can be transferred through the network
 - XML for structural data
 - binary and other structural formats (JSON) also permitted
 - HTTP methods directly used to manipulate resources
 - GET, PUT, DELETE - obvious semantics
 - other HTTP methods, HTTP authentication, cookies, additional headers and arguments - all may be used to implement additional features

REST for XML database

- REST – remote access to a repository
 - Can it be an XML database? Why not...
- Possible applications:
 - Access API independent of particular platform or pr.lang.
 - Easy and efficient remote access from
 - Javascript clients (AJAX)
 - mobile clients
 - Integration with XML-related standards
 - XSLT, XQuery – documents available through HTTP URLs
 - XForms – acquiring and modifying documents directly form XForms
- HTTP interface available also to call server-side XQuery scripts
- XRX architecture: XForms + REST + XQuery

XForms

- XML application for specification of interactive forms
- Versions:
 - 1.0 – 2003
 - 1.1 – 2009 (currently most commonly used)
 - 2.0 – WD
- More than HTML forms:
 - data model defined separately from UI
 - by example or using XML Schema
 - processing model specified with events and actions
 - XPath used to tie data instance with UI controls and actions
 - various data access modes given in submission module
 - including REST-compliant HTTP access
 - more UI controls, interactive switch, automatic repeat

XForms – document structure

- Forms are embedded in a host document, usually XHTML
- Data model – `xf:model` element
 - anywhere in host document
 - header more elegant
 - but body more practical for dynamic documents
 - more than one model available; in such case they must have identifiers
- Form controls (in XForms namespace)
 - placed within normal XHTML tags
 - (some of them) may contain further XHTML fragments
- Action specifications and constraints tied with XForm elements
 - by inserting them inside model fragments or control tags
 - using general `xf:bind` elements