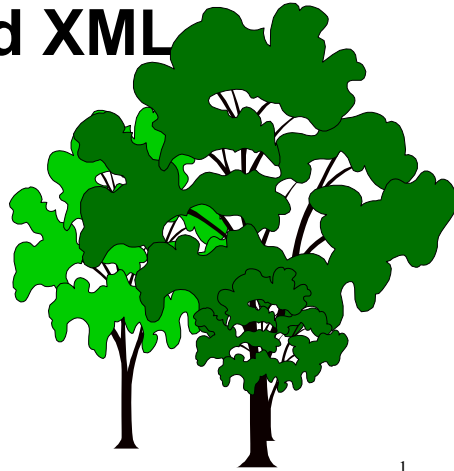


JBuilder 5 und XML

Michael Schmut
Thomas Forster

schmut@oio.de

Orientation in Objects GmbH
Weinheimer Str. 68
68309 Mannheim
www.oio.de

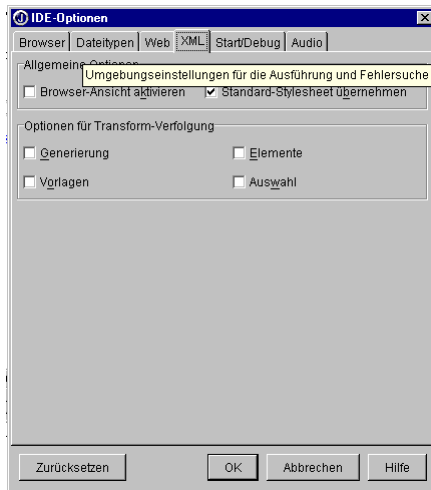


1

Inhalt

- **XML - Konfiguration von JBuilder**
- **XML - Umwandlung und Validierung**
- **Web - Publishing mit Cocoon**
- **XML - Databinding mit Castor und BorlandXML**
- **XML - Datenbankkomponenten im JBuilder**

2



- **Browseransicht aktivieren :**
macht den integrierten Browser verfügbar
- **Standardstylesheet übernehmen :**
zeigt die Baumansicht einer XML -Datei
- **Transform - Verfolgung :**
Optionen für die Transform-Verfolgung mit deren Hilfe bei einer Umwandlung der Programmfluß im Meldungsfenster verfolgt werden kann , wenn ein Stylesheet angewendet wird.

3

Inhalt

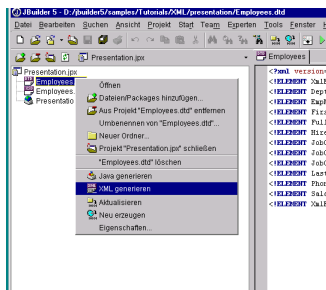
- XML - Konfiguration von JBuilder
- **XML - Umwandlung und Validierung**
- Web - Publishing mit Cocoon
- XML - Databinding mit Castor und BorlandXML
- XML - Datenbankkomponenten im JBuilder

4

- Integration des Xerces - Parsermoduls
- DTD zu XML Experte
- XML zu DTD Experte
- Validierung gegen Schemata und DTDs
- XML – Betrachter (Browser)

5

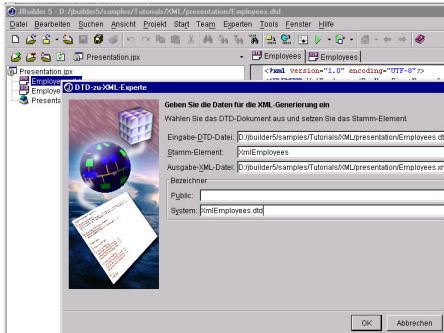
Erstellen eines XML - Files aus einer DTD



- Rechtsklick auf die DTD
- „XML - Erstellen“ auswählen

6

Erstellen eines XML - Files aus einer DTD



- Root - Element auswählen
- Ausgabe - Datei angeben
- im Feld „System“ die zu referenzierende DTD angeben

7

Inhalt

- XML - Konfiguration von JBuilder
- XML - Umwandlung und Validierung
- **Web - Publishing mit Cocoon**
- XML - Databinding mit Castor und BorlandXML
- XML - Datenbankkomponenten im JBuilder

8

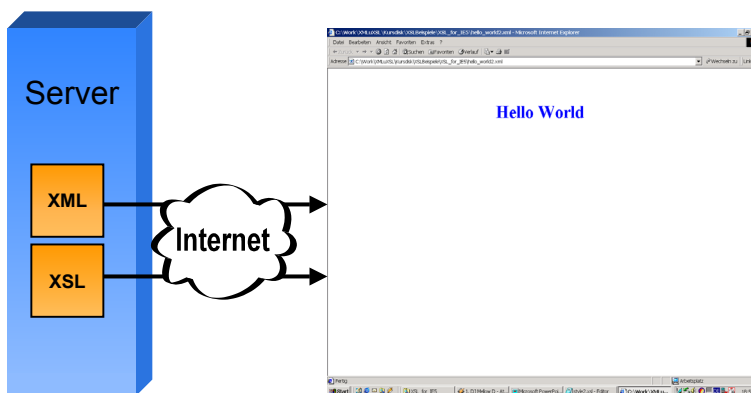
Webpublishing

- Starres Layout
- Großer Änderungsaufwand
- Aufwand im Contentmangement

Webentwicklung

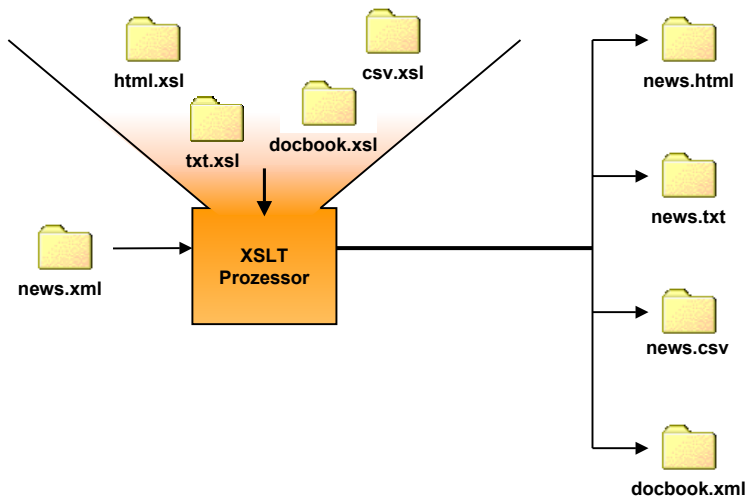
- Unübersichtliche ServerPage Anwendungen
- Problematische Trennung der Kompetenzen und Aufgaben
- Starres Layout

9

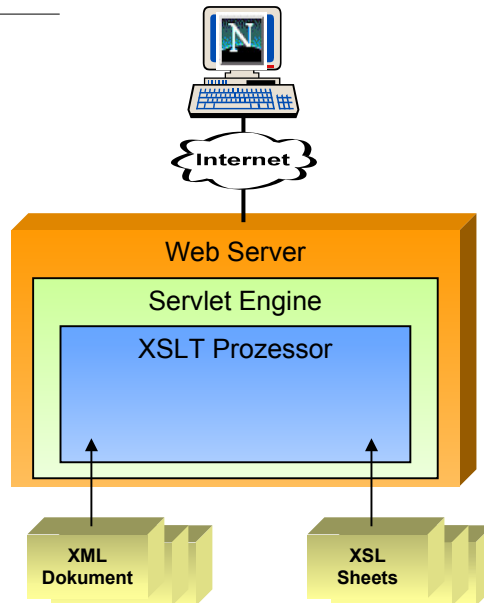


10

- Server wandelt XML in HTML, WML,... um
- Aufbereitung für verschiedene Clients ist möglich
 - IE, Netscape, WAP-Client, ...
- Keine speziellen Anforderungen an den Browser
 - Java, Neue Version, ...



Idee: XSLT im Servlet



13



- Apache Cocoon is a 100% pure Java publishing framework that relies on new W3C technologies (such as DOM, XML, and XSL) to provide web content.

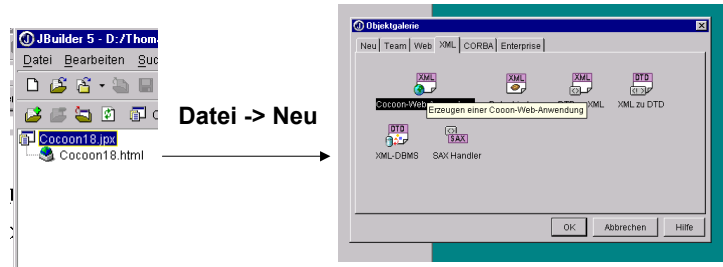
Quelle: Cocoon 1 README

A small revolution

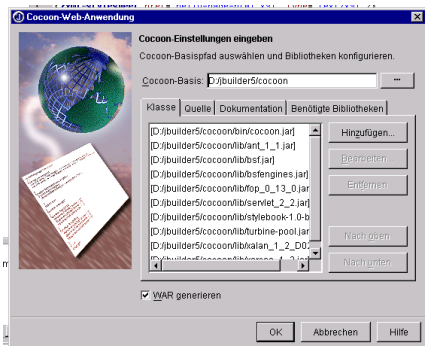
Stefano Mazzocchi (Autor und Architekt von Cocoon)

14

Cocoon in ein Projekt einbinden :



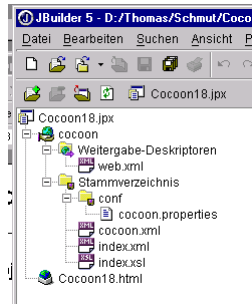
Cocoon in ein Projekt einbinden :



Konfiguration der Bibliotheken

- Cocoon - Basis
- Klasse
- Quelle
- Dokumentation
- Bibliotheken
- WAR generieren

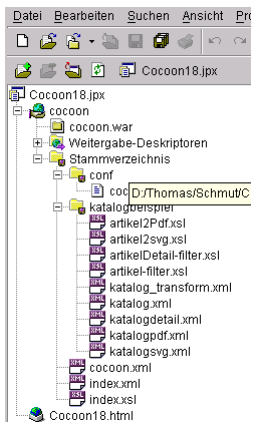
Projektfenster :



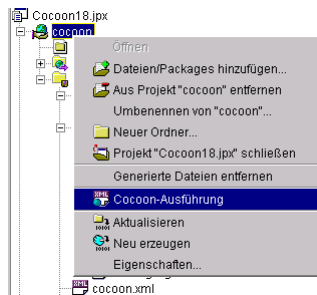
Dateien, die vom Cocoon - Experten erstellt werden

- Cocoon.war
- web.xml
- cocoon.properties
- cocoon.xml
- index.xml/xsl

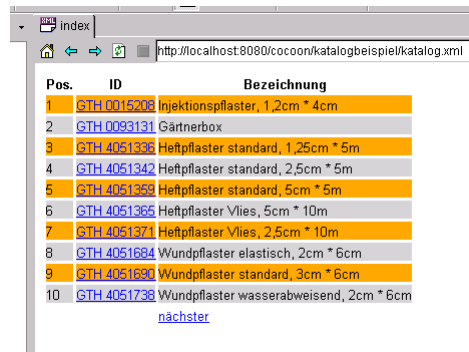
Dateien in das Stammverzeichnis kopieren :



Cocoon ausführen :



Der Browser :



The screenshot shows a web browser window with the address bar containing `http://localhost:8080/cocoon/katalogbeispiel/katalog.xml`. The page content is a table with 10 rows. The first 9 rows are highlighted in yellow, and the 10th row is highlighted in blue. The table has three columns: 'Pos.', 'ID', and 'Bezeichnung'. Below the table, there is a blue link labeled 'nächster'.

Pos.	ID	Bezeichnung
1	GTH 0015206	Injektionspflaster, 1,2cm * 4cm
2	GTH 0093131	Gärtnerbox
3	GTH 4051336	Heftpflaster standard, 1,25cm * 5m
4	GTH 4051342	Heftpflaster standard, 2,5cm * 5m
5	GTH 4051369	Heftpflaster standard, 5cm * 5m
6	GTH 4051366	Heftpflaster Vlies, 5cm * 10m
7	GTH 4051371	Heftpflaster Vlies, 2,5cm * 10m
8	GTH 4051684	Wundpflaster elastisch, 2cm * 6cm
9	GTH 4051690	Wundpflaster standard, 3cm * 6cm
10	GTH 4051738	Wundpflaster wasserabweisend, 2cm * 6cm

[nächster](#)

19

Inhalt

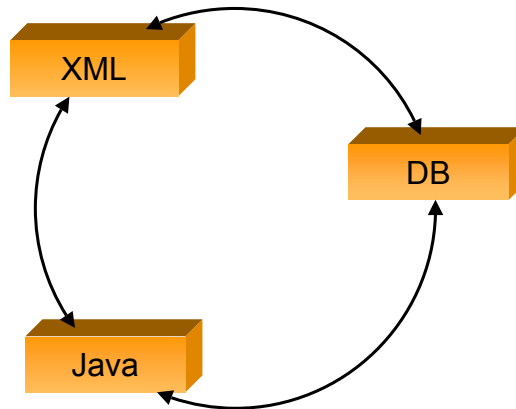
- XML - Konfiguration von JBuilder
- XML - Umwandlung und Validierung
- Web - Publishing mit Cocoon
- XML - Databinding mit Castor und BorlandXML
- XML - Datenbankkomponenten im JBuilder

20

- Mit Hilfe von Datenbindung kann man auf Daten zugreifen, diese Bearbeiten und die bearbeiteten Daten an eine Datenbank zurückschicken oder mit einem XML – Dokument anzeigen.
- Das XML –Dokument fungiert als Übertragungsmechanismus zwischen Anwendung und Datenbank.

- Die Datenbindung wird implementiert, indem Java Klassen generiert werden, die für die in einer Grammatik (DTD o. Schema) enthaltenen Begrenzungen stehen.
- Die XML – Files entsprechen dann den Instanzen der Klassen.
- Mit den so erstellten Klassen lassen sich XML – Dokumente lesen, erstellen oder validieren, die dieser Grammatik entsprechen.

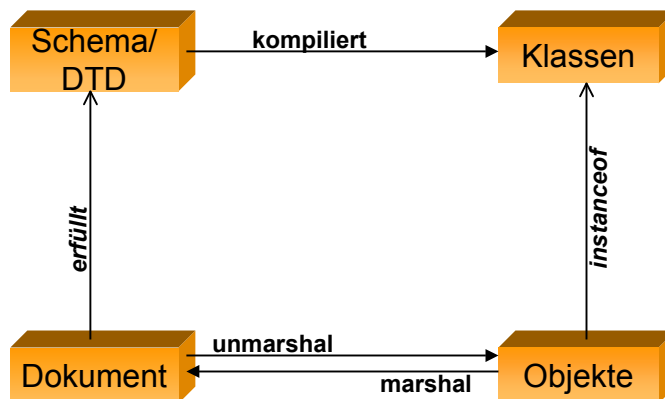
XML - Data - Binding



23

XML - Data - Binding

„Roundtrip“



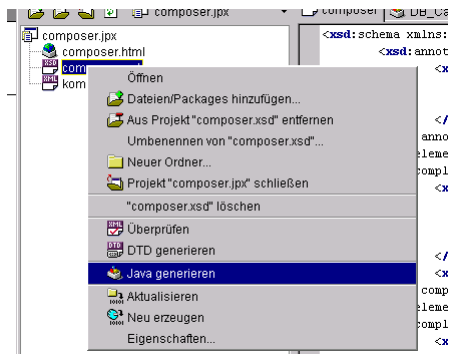
24

Das Objektmodell :

- Castor ist ein Datenbindungssystem, das eine Instanz eines XML – Schemas einem Objekt zuordnet das für die Daten steht.
- Das Objektmodell umfaßt Klassen, Typen sowie Deskriptoren, die verwendet werden, um Infos zu einer Klasse und deren Feldern abzurufen.

Marshalling :

- Marshalling beschreibt einfach einen Vorgang, bei dem ein Objekt in einen Datenstrom umgewandelt wird und umgekehrt.
- Das Castor – System verwendet ein Marshalling – System, das ClassDescriptors und FieldDescriptors verwendet, um das Marhalling von und nach XML zu beschreiben.



- Rechtsklick auf das Schema
- „Java generieren“ auswählen

27

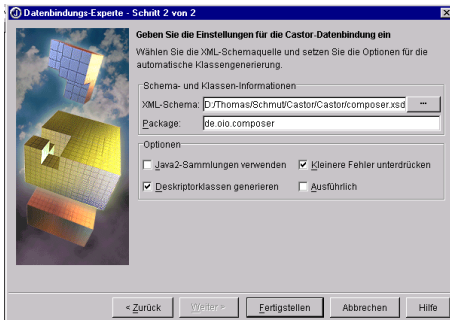
Der Datenbindungsexperte :



- Typ der Datenbindung „Castor“
- „Weiter“

28

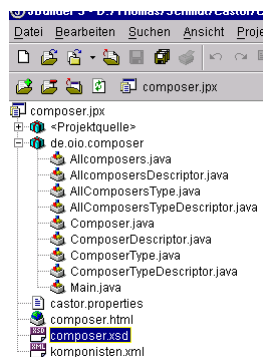
Der Datenbindungsexperte für Castor:



- Pfad des Schemas wird vorbesetzt
- Package - Namen angeben
- evtl. weitere Optionen festlegen
- „Fertigstellen“

29

Projektansicht :

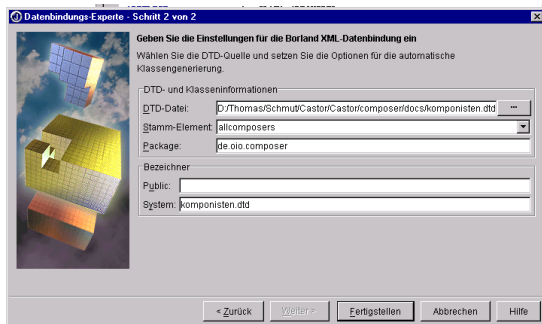


- Java - Dateien, die vom Datenbindungsexperten erstellt wurden

30

- BorlandXML verwendet DTDs um Java – Klassen zu generieren.
- Schritt 1 : Generierung einer Klassenmodelldatei aus einer DTD.
Die Datei beschreibt eine übersichtliche Struktur der Zielklassen und ermöglicht es diese Klassen anzupassen
- Schritt 2 : BorlandXML generiert nun Java – Klassen aus der Klassenmodell – Datei .

Der Datenbindungsexperte für BorlandXML:



Funktionalität von BorlandXML

- JavaBean – Bearbeitung um ein XML – Dokument zu erstellen oder auf Daten im Dokument zuzugreifen.
- Marshalling und Unmarshalling .
- Dokumentenvalidierung : Überprüfen von JavaBean – Objekten vor dem Marshalling zu XML bzw. nach dem Unmarshalling eines Dokumentes zu JavaBean – Objekten .

- **XML - Konfiguration von JBuilder**
- **XML - Umwandlung und Validierung**
- **Web - Publishing mit Cocoon**
- **XML - Databinding mit Castor und BorlandXML**
- **XML - Datenbankkomponenten im JBuilder**

Typen von XML Dokumenten

- Datenzentrische XML - Dokumente
- Dokumentenzentrische XML -Dokumente

datenzentrische XML - Dokumente

- XML dient als Transportmedium zwischen DB und Anwendung
- Daten werden zur maschinellen Weiterverarbeitung verwendet
- sehr hohe Granularität der Daten
- regelmäßige Struktur
- keine Mischinhalte

datenzentrisches XML - Dokument

```
<SalesOrder SONumber="12345">  
  <Customer CustNumber="543">  
    <CustName>ABC Industries</CustName>  
    <Street>123 Main St.</Street>  
    <City>Chicago</City>  
    <State>IL</State>  
    <PostCode>60609</PostCode>  
  </Customer>  
  ....  
</SalesOrder>
```

37

dokumentenzentrische XML - Dokumente

- eher unregelmäßige Struktur
- weniger granuliert
- viel Mischinhalt
- die Reihenfolge des Auftauchens der PCDATA und Childelemente ist wichtig

38

dokumenten-zentrisches XML - Dokument

```
<Produkt>
  <Name>Bananenquark</Name>
  <Hersteller>Frutti GmbH Hinterdupfingen</Hersteller>
  <Kurzbeschreibung>leckere Bananenquark aus dem Hinterland</Kurzbeschreibung>
  <Beschreibung>
    <Para>Der leckere Bananenquark wird in <i>farbigen Plastikschaalen</i>, die <b>
      völlig ökologisch</b> gereinigt wurden, ausgeliefert.
      Die verwendeten Bananen wurden aus den <b>
        besten Bananen Südschwedens</b> ausgewählt. ....
    </Para>
    <Para>Sie können:</Para>
    <Liste>
      <Eintrag>
        <Link dest="bestellen.html">gleich Bananenquark bestellen </Link>
      </Eintrag>
      <Eintrag>
        <Link dest="schweden.html">mehr über den Auswahlprozess der Bananen
          in Schweden erfahren </Link>
      </Eintrag>
    </Liste>
  </Beschreibung>
  ....
</Produkt>
```

39

Dokumente in die Datenbank bringen

Software, die Daten zwischen Dokumenten und relationalen Datenbanken austauscht, benötigt ein sogenanntes „Mapping“. Hierbei unterscheidet man zwischen zwei Formen:

- Vorlagenbasiertes Mapping (Template - Driven Mapping)
- Modellbasiertes Mapping (Model - Driven Mapping)

40

- Es gibt kein vordefiniertes mapping zwischen Dokument und Datenbank
- produktspezifische Kommandos werden in eine Vorlage eingebettet, die von der Übertragungssoftware ausgeführt werden
- Die Middleware ersetzt die Anweisung durch die korrespondierenden Daten und formatiert das Ganze als XML

Vorlage :

```
<FlightInfo>
  <Introduction>The following flights have available seats:</Introduction>
  <SelectStmt>SELECT Airline,      FltNumber, Depart, Arrive FROM Flights</SelectStmt>
  <Conclusion>We hope one of these meets your needs</Conclusion>
</FlightInfo>
```

Rückgabe :

```
<FlightInfo>
  <Introduction>The following flights have available seats:</Introduction>
  <Flights>
    <Row>
      <Airline>ACME</Airline>
      <FltNumber>123</FltNumber>
      <Depart>Dec 12, 1998 13:43</Depart>
      <Arrive>Dec 13, 1998 01:21</Arrive>
    </Row> ...
  </Flights>
  <Conclusion>We hope one of these meets your needs.</Conclusion>
</FlightInfo>
```

- Daten entsprechen einer vordefinierten Form und werden dementsprechend „gemapped“
- Gegenüber dem template-driven Mapping wird Flexibilität eingebüßt, allerdings wird die Einfachheit erhöht.
- Dokumente, die nicht der vordefinierten Form entsprechen, können per XSLT transformiert werden.

Map - File :

```
<XMLToDBMS Version="1.0">
  <Options>
  </Options>
  <Maps>
  <ClassMap>
    <ElementType Name="composer"/>
    <ToClassTable>
    <Table Name="composer"/>
    </ToClassTable>
    <PropertyMap>
      <Attribute Name="epoche"/>
      <ToColumn>
        <Column Name="epoche"/>
      </ToColumn>
    </PropertyMap>
    <PropertyMap>
      <ElementType Name="name"/>
      <ToColumn>
        <Column Name="name"/>
      </ToColumn>
    </PropertyMap>
  </ClassMap>
  </Maps>
</XMLToDBMS>
```

Sichtweisen der Abbildung von Daten in einem XML - Dokument :

- als Tabellenmodell und daraus resultierend das tabellenbasierte Mapping
- als Objektmodell und daraus resultierend das objekt – relationale Mapping

- Wird oft von Middleware Produkten benutzt, um Daten zwischen XML Dokumenten und relationalen Datenbanken auszutauschen.
- XML Dokumente werden als einzelne oder mehrere Tabellen in der Datenbank modelliert.
- Die XML - Dokumente müssen einer bestimmten Struktur genügen

Tabellenbasiertes Mapping

Struktur eines XML - Files und die entsprechende Tabelle:

```
<A>
  <B>
    <C>ccc</C>
    <D>ddd</D>
    <E>eee</E>
  </B>
  <B>
    <C>fff</C>
    <D>ggg</D>
    <E>hhh</E>
  </B>
</A>
```

	C	D	E
<=>
	ccc	ddd	eee
	fff	ggg	hhh

47

Objekt - Relationales Mapping

- Wird von der XML -DBMS Middleware im JBuilder verwendet
- Beim Objekt - Relationalen Mapping werden die Daten des XML Dokumentes als Baum von Objekten modelliert

48

Objekt - Relationales Mapping

XML - File, Objektmodell und Tabellenmodell

```
<SalesOrder>
  <Number>1234</Number>
  <Customer>Gallagher Industries</Customer>
  <Date>29.10.00</Date>
  <Item Number="1">
    <Part>A-10</Part>
    <Quantity>12</Quantity>
    <Price>10.95</Price>
  </Item>
  <Item Number="2">
    <Part>B-43</Part>
    <Quantity>600</Quantity>
    <Price>3.99</Price>
  </Item>
</SalesOrder>
```

```
object SalesOrder {
  number = 1234;
  customer = "Gallagher Industries";
  date = 29.10.00;
  items = {ptrs to Item objects};
}

object Item {
  number = 1;
  part = "A-10";
  quantity = 12;
  price = 10.95;
}

object Item {
  number = 2;
  part = "B-43";
  quantity = 600;
  price = 3.95;
}
```

SaleOrders

Number	Customer	Date
1234	Gallagher Industries	29.10.00
...

Items

SONumber	Item	Part	Quantity	Price
1234	1	A-10	12	10.95
1234	2	B-43	600	3.99
...

49

Vorlagenbasierte Komponenten in JBuilder

JBuilder kommt mit zwei vorlagenbasierten Komponenten, XTable und XQuery. Die Komponenten sind Unidirektional, es können nur Daten aus der Datenbank herausgezogen werden.

Die Parametrisierung dieser Komponenten kann auf drei Arten erfolgen :

- über den Customizer der Komponente
- über den Inspektor
- mit Hilfe eines XML - Abfragedokumentes

50

XTable :

- führt eine Select-Anweisung auf eine Untermenge einer Tabelle aus.
- Die Untermenge ergibt sich aus der Bedingung (Schlüssel-Wert-Paare), die in der where-Klausel der Select-Anweisung definiert ist.
- Bei der Ausführung wird eine Ergebnismenge in XML generiert.
- Der Tabellename, Schlüssel-Wert-Paare und andere Parameter werden entweder in einem XML-Abfragedokument, in Bean-Eigenschaften oder in beidem bereitgestellt

51

XQuery :

- Führt eine SQL-Anweisung aus und generiert eine Ergebnismenge in XML.
- SQL und weitere Parameter werden entweder in einem XML-Abfragedokument, in Bean-Eigenschaften oder in beidem bereitgestellt.
- der Unterschied zu XTable besteht darin, dass die Komponente direkt eine SQL - Anweisung entgegennimmt.

52

Modellbasierte Komponenten in JBuilder



JBuilder implementiert dazu die Open - Source Middleware XML-DBMS.
Diese Lösung benötigt folgende Komponenten:

- relationale Datenbank mit einem JDBC - Treiber
- XML - Dokument zur ein und Ausgabe von Daten
- Map - File
- API - Methoden zu Datenübertragung von und zur Datenbank

53

Modellbasierte Komponenten in JBuilder



JBuilder bietet folgende XML - DBMS - Unterstützung :

- XML - DBMS - Experte
- die Komponente XMLDBMSTable
- die Komponente XMLDBMSQuery

54

Der XML - DBMS Experte :

- Erstellen einer SQL - Script - Datei auf Grundlage einer DTD
- Erstellen einer Map - Datei
- noch keine Unterstützung zu Erstellung von Map - Dateien aus Datenbankschemata

XMLDBMSQuery und XMLDBMSTable :

- Diese Beans werden verwendet, um Datenbankdaten in ein XML-Dokument bzw. XML - Dokumente in Datenbankdaten umzuwandeln.
- Sie werden in Verbindung mit der MAP-Datei und Datenbanktabellen verwendet, die durch den Experten zum Generieren von XML-DBMS-Mapping und SQL erzeugt wurden.
- der Unterschied zwischen den Komponenten besteht darin, dass die XMLDBMSQuery - Komponente eine SQL - Anweisung entgegennimmt und nur Unidirektional ist.

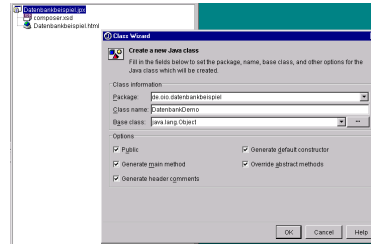
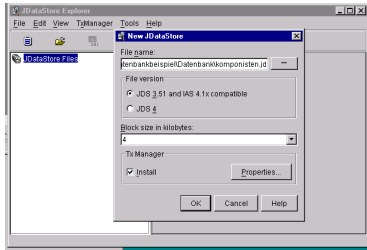
Beispiel zu XMLDBMSQuery und XMLDBMSTable



Orientation in Objects

Neue Datenbank anlegen :

Klasse zum Projekt hinzufügen :



57

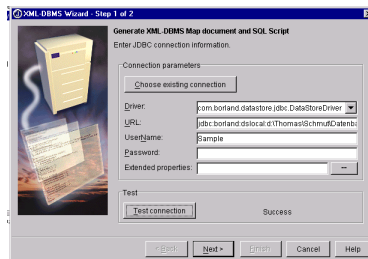
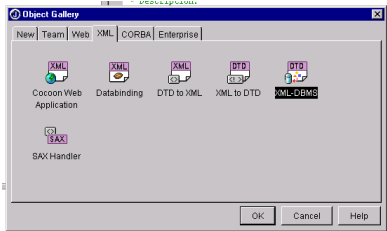
Beispiel zu XMLDBMSQuery und XMLDBMSTable



Orientation in Objects

XML DBMS - Experten aufrufen :

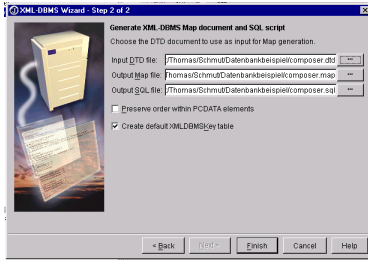
JDBC - Einstellungen :



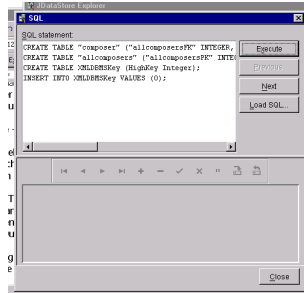
58

Beispiel zu XMLDBMSQuery und XMLDBMSTable

Map und SQL - File erzeugen :



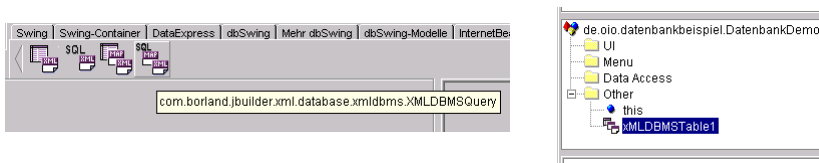
Tabellen erzeugen :



59

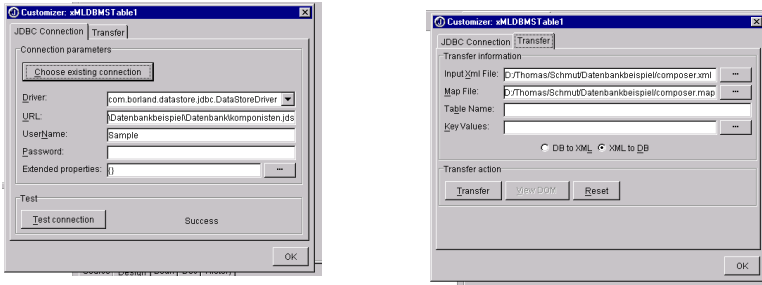
Beispiel zu XMLDBMSQuery und XMLDBMSTable

XMLTable - Bean über den Designer hinzufügen :

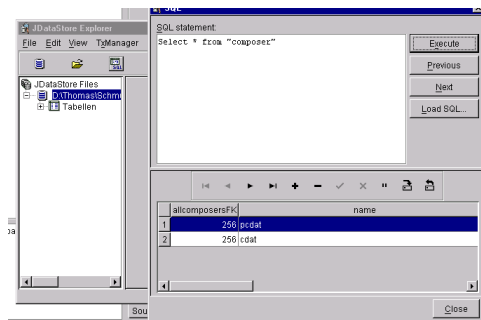


60

XMLTable - Bean über den Customizer konfigurieren :



Überprüfen, ob die Übertragung funktioniert hat :





Bookmarks :

Seybold Publications and O'Reilly & Associates, Inc :

- www.xml.com

Dokumentation zu XML - DBMS

- www.rpbouret.com

JBuilder5 :

- www.borland.com
- JBuilder5 Online - Hilfe

Cocoon Home :

- xml.apache.org

Castor Home :

- castor.exolab.org

Ende

**Danke für Ihre
Aufmerksamkeit!**