

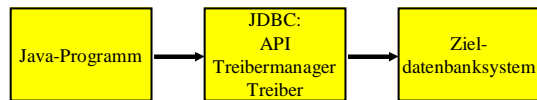
Datenbank-Grundlagen

SS 2005
Kapitel 8: Datenbank-Einbettung
in Programmiersprachen

Prof. Dr. Stefan Böttcher
Universität Paderborn

Java Database Connectivity (JDBC)

- Paket von Java-Klassen zum DB-Zugriff mit SQL
- vom Ziel-DBMS unabhängige API
- Standard seit Java 1.1



Java-Interfaces von JDBC

- **SQLDriver** : Treiber für ein Ziel-DBMS oder ODBC
- **SQLDriverManager** : registriert Treiber
- **Connection** : Für Verbindungen
- **Statement** : für Statement-Objekt, z.B. Query
- **ResultSet** : für Ergebnismenge

Die Klassen zu diesen Interfaces werden von DB-Herstellern implementiert
JDBC-ODBC-Brücke wird von SUN mitgeliefert

Datenbank anschließen unter ODBC

1. DB da
2. ODBC-Name hinzufügen
3. OK wählen

Datenbankzugriffe mit JDBC

- **Treiber laden**
`Class c = Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");`
- **Verbindung herstellen**
`con = DriverManager.getConnection("jdbc:odbc:odbc2access");`
- **Statement-Objekte definieren**
`Statement stmt = con.createStatement();`
- **Datenbank zugreifen, z.B. einfügen**
`stmt.executeUpdate("insert into Liefert values('IBM', 'pc500', 2500,6)");`
- **Statement und Verbindung zum DBMS schließen**
`stmt.close(); con.close();`

Datenbank-Anfragen mit JDBC

```
Class c = Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");  
con = DriverManager.getConnection("jdbc:odbc:odbc2access");  
Statement stmt = con.createStatement();
```

Datenbankanfrage stellen

```
ResultSet rsLiefert =  
    stmt.executeQuery(" select * from Liefert where Teil = 'pc500' ");  
  
while ( rsLiefert.next() ) // hole nächstes Tupel aus Result-Set  
{  
    ausgabe += rsLiefert.getString("Lieferant");  
    // ggf. weitere Spalten ausgeben  
}  
  
stmt.close(); con.close();
```

Datenbank anlegen mit JDBC-Programm

```
import java.sql.*;
public class dbinit
{
    public static void main( String[] args )
    {
        String ergebnis = "";
        try {
            Class c = Class.forName("sun.jdbc.odbc.JdbcOdbcDriver"); // Treiber für ODBC
            Connection con = DriverManager.getConnection("jdbc:odbc:odbc2access");
            try {
                ergebnis = makeDB( con ); // : :DB-Name unter ODBC
                System.out.println( ergebnis );
            } finally { con.close(); }
        } catch (Exception e) { System.out.println( e ); }
    } // main zuende

    public static String makeDB( Connection con )
    {
        String ausgabe=""; // String zum Sammeln der Ausgabe
        try {
            Statement stmt = con.createStatement();
            stmt.executeUpdate(
                "create table Liefert( Lieferant char(10), Teil char(10), Preis int, Lieferzeit int ) " );
            stmt.executeUpdate( "insert into Liefert values('Vobis','pc400',1700,3)" );
            stmt.close(); // Statement schließen
            ausgabe += "\nDatenbank initialisiert.\n";
        } catch (Exception e) { ausgabe += "\n" + "Fehler: " + e ; }
        return ausgabe ;
    } // makeDB zuende
} // class dbinit zuende
```

Datenbank-Grundlagen SS 2005 Prof. Dr. Stefan Böttcher Folie JDBC/7

Datenbank lesen mit JDBC-Programm (1)

```
import java.sql.*;
public class dbtab
{
    public static void main( String[] args )
    {
        String ergebnis = "";
        try {
            Class c = Class.forName("sun.jdbc.odbc.JdbcOdbcDriver"); // Treiber für ODBC
            Connection con = DriverManager.getConnection("jdbc:odbc:odbc2access");
            try {
                ergebnis = selectTab( con, "2200" ); // : :DB-Name unter ODBC
                System.out.println( ergebnis );
            } finally { con.close(); }
        } catch (Exception e) { System.out.println( e ); }
    } // main zuende

    // es folgt noch :
    public static String selectTab( Connection con, String limit )
    {
        // berechne die richtige Ausgabe → siehe nächste Folie
        // Funktion selectTab zuende
    } // class dbselect zuende
```

Datenbank-Grundlagen SS 2005 Prof. Dr. Stefan Böttcher Folie JDBC/8

Datenbank lesen mit JDBC-Programm (2)

```
import java.sql.*;
public class dbtab
{
    public static void main( String[] args ) { ... }

    public static String selectTab( Connection con, String limit )
    {
        String ausgabe=""; // String zum Sammeln der Ausgabe
        try {
            Statement stmt = con.createStatement();
            ResultSet rsLiefert = stmt.executeQuery(
                "SELECT * FROM Liefert WHERE Preis < " + limit );
            // Strings in SQL müßten zusätzlich in einfache Hochkommas:
            // "SELECT * FROM Liefert WHERE Teil = " + limit + "" );
            ausgabe += "\n\nLiefert:\n( Lieferant Teil Preis Lieferzeit ) ";

            while (rsLiefert.next()) // hole nächstes Tupel aus Result-Set
            {
                ausgabe += "\n" + rsLiefert.getString("Lieferant") +
                    " " + rsLiefert.getString("Teil") +
                    " " + rsLiefert.getInt("Preis") +
                    " " + rsLiefert.getInt("Lieferzeit") ;
            }
            rsLiefert.close(); stmt.close(); // ResultSet schließen, Statement schließen
        } catch (Exception e) { ausgabe += "\nFehler bei Anfrage an die Datenbank:\n" + e ; }
        return ausgabe ;
    } // Funktion selectTab zuende
} // class dbselect zuende
```

Datenbank-Grundlagen SS 2005 Prof. Dr. Stefan Böttcher Folie JDBC/9

Datenbankschema lesen

```
import java.sql.*;
public class dbinit
{
    public static void main( String[] args )
    {
        String ergebnis = "";
        try {
            Class c = Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            Connection con = DriverManager.getConnection("jdbc:odbc:odbc2access");
            ergebnis = accessDB( con );
            System.out.println( ergebnis );
        } catch (Exception e) { System.out.println( e ); }
    }

    public static String accessDB( Connection con )
    {
        String ausgabe=""; // String zum Sammeln der Ausgabe
        try {
            DatabaseMetaData md = con.getMetaData();
            final String[] tabellen = {"TABLE"}; // Hilfsvariable
            ResultSet tablesNames = md.getTables( null, null, null, tabellen );
            while (tablesNames.next())
            {
                String tablename = new String(tablesNames.getString(3));
                ausgabe += tablename + "\n";
            }
        } catch (Exception e) { ausgabe += e ; }
        return ausgabe;
    } // accessDB
} // dbinit
```

Metadaten der Datenbank lesen

Tabellennamen der Datenbank lesen

Datenbank-Grundlagen SS 2005 Prof. Dr. Stefan Böttcher Folie JDBC/10

Tabelle komplett lesen

```
import java.sql.*;
public class dbtab
{
    // Main-Funktion wie in den anderen Programmen, jedoch Aufruf:
    // ergebnis = accessTab( con, "Auftrag" );

    public static String accessTab( Connection con, String tabelle )
    {
        int spalte;
        String ausgabe=""; // String zum Sammeln der Ausgabe
        try {
            Statement stmt = con.createStatement();
            ResultSet rs = stmt.executeQuery("select * from " + tabelle);
            ResultSetMetaData rsm = rs.getMetaData();
            int spaltenAnzahl = rsm.getColumnCount();
            for( spalte=1 ; spalte <= spaltenAnzahl ; spalte++)
            {
                ausgabe += rsm.getColumnLabel( spalte ) + "\t\t";
            }
            ausgabe += "\n\n";
            while (rs.next())
            {
                for( spalte=1 ; spalte <= spaltenAnzahl ; spalte++)
                {
                    ausgabe += rs.getString(spalte) + "\t\t";
                }
                ausgabe += "\n";
            }
        } catch (Exception e) { ausgabe += e ; }
        return ausgabe;
    } // accessTab
} // dbtab
```

Metadaten der Tabelle lesen

Datenbank-Grundlagen SS 2005 Prof. Dr. Stefan Böttcher Folie JDBC/11

Zugriff auf Oracle-Datenbanken

- **Treiber laden**
 Class c = Class.forName("oracle.jdbc.OracleDriver");
 // Treiber für Oracle
- **Verbindung herstellen**
 Connection con = DriverManager.getConnection(
 "jdbc:oracle:thin:@131.234.48.168:1521:oradb01",
 "oracle_login", "oracle_passwort");
 Internetadresse Port Datenbank
- **Alles weitere wie für Access:**
 Statement-Objekt definieren, Datenbank zugreifen, Statement und Verbindung zum DBMS schließen

Datenbank-Grundlagen SS 2005 Prof. Dr. Stefan Böttcher Folie JDBC/12

Zugriff auf Sybase-Datenbank (unter Unix)

- Treiber laden

```
Class c = Class.forName("com.sybase.jdbc.SybDriver");
```
- Verbindung herstellen

```
con = DriverManager.getConnection(
"jdbc:sybase:Tds:beethoven.uni-paderborn.de:4100/kunden",
"userid", "password");
```

↑ Internetadresse
↑ Port
↑ Datenbank
- Alles weitere wie für Access:
Statement-Objekt definieren, Datenbank zugreifen, Statement und Verbindung zum DBMS schließen

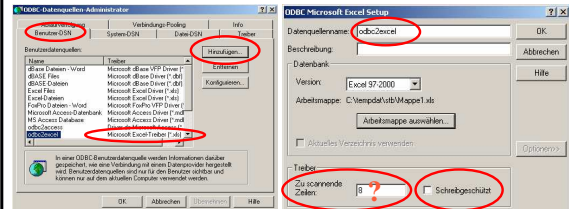
Datenbank-Grundlagen SS 2005 Prof. Dr. Stefan Böttcher Folie JDBC/13

Zugriff auf Excel-Tabellen

- Treiber laden

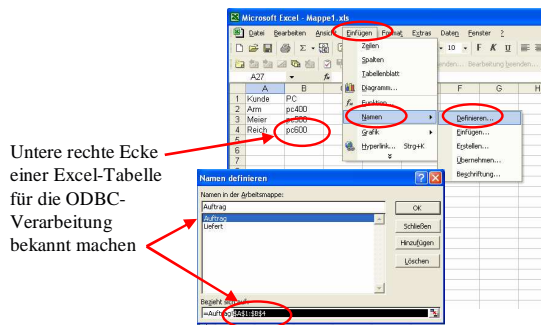
```
Class c = Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```
- Verbindung herstellen

```
con = DriverManager.getConnection("jdbc:odbc:odbc2excel");
```



Datenbank-Grundlagen SS 2005 Prof. Dr. Stefan Böttcher Folie JDBC/14

Zugriff auf Excel-Tabellen



Datenbank-Grundlagen SS 2005 Prof. Dr. Stefan Böttcher Folie JDBC/15

Beispielprogramme zu JDBC

- dbinit.java Datenbank-Initialisierung
 - dbselect.java Selektion von Datenbankinhalten
 - dbinfo.java Information über das Datenbankschema lesen nutzt Metadaten der Datenbank
 - dbtbl.java Eine beliebige Tabelle ausgeben nutzt Metadaten der auszugebenden Tabelle
 - exinit.java, ..., extab.java, orainit.java, sybinit.java dasselbe für Excel, Oracle und Sybase
- Quellcode in .zip-Datei

Datenbank-Grundlagen SS 2005 Prof. Dr. Stefan Böttcher Folie JDBC/16

Prepared Statements

- Statt

```
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery(
"select * from liefert where teil = " + pc + " ");
```
- wird folgendes verwendet

```
PreparedStatement pstmt = con.prepareStatement(
"select * from liefert where teil = ?");
pstmt.setString(1, pc);
ResultSet rs = pstmt.executeQuery();
```
- Vorteile:
- effizienter verarbeitbar durch DBMS
 - verhindert SQL-Injection, d.h. Übergabe pc="pc1' union select login, password from user"

Datenbank-Grundlagen SS 2005 Prof. Dr. Stefan Böttcher Folie JDBC/17