Pemrograman Berorientasi Obyek

anton@ukdw.ac.id

JAR dan JDBC

JAR File

- File JAR adalah **file executable** dari class – class Java.
- File JAR berfungsi membungkus *satu atau lebih file-file class berserta informasinya* menjadi sebuah file archive.
- File JAR dikompresi dengan metode **ZIP** sehingga kita dapat membukanya dengan program-program kompresi seperti WinZip atau WinRAR.
- Dalam Java juga disediakan program **utility tools** JAR yang berfungsi untuk memanipulasi file-file JAR.

Keuntungan JAR

- Decrease download time: karena file JAR ukurannya kecil dan biasanya terdiri dari satu buah file saja.
  - Jadi proses download akan lebih mudah dan cepat.
- Compressed
  - Karena file JAR secara otomatis mengkompres file-file Java, sehingga memperkecil ukuran file.
- *Package Versioning and Information*
  - File JAR dapat diberi informasi tertentu yang unik. Misalnya informasi versi, vendor, main-class dan lain-lain.
- *Portability*
  - Karena file JAR akan dapat diproses oleh JRE yang bersifat multiplatform

Manipulasi JAR

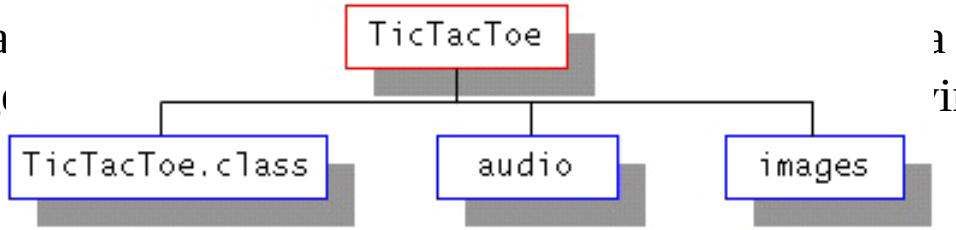
- Membuat JAR
  - Sintaks :
    - jar –cvf <namafileJAR> <namafileclass1> [<namafileclass2>,<namafileclass3>,…]
  - Contoh : jar –cvf coba.jar a.class b.class c.class (c: create)
  - Contoh : jar –cvf coba2.jar *.class
- Melihat isi JAR
  - Sintaks : jar –tvf <namafileJAR> (t: table of contents)
  - Contoh : jar –tvf coba.jar
- Mengekstrak JAR
  - Sintaks : jar –xvf <namafileJAR> (x: extract)
  - Contoh : jar –xvf coba.jar
- Mengupdate JAR
  - Sintaks: jar –uvf <namafileJAR> <input-files> (u:update)

Manipulasi JAR

- Mengekstrak satu atau beberapa file dalam JAR
  - Sintaks: jar –xvf jar-file filename(s)
- Menjalankan aplikasi java dalam JAR
  - Butuh Main-Class pada **Manifest**
  - Sintaks: java –jar jar-file
- Menjalankan file jar pada **applet**
  - <applet code=AppletClassName.class
    archive="JarFileName.jar"
    width=width height=height>
    </applet>

TicTacToe Applet

The JDK demos include a bytecode class file, audio files, and images using this structure:



The audio and images subdirectories contain sound files and GIF images used by the applet.

To package this demo into a single JAR file named TicTacToe.jar, you would run this command from inside the TicTacToe directory:

```
jar -cvf TicTacToe.jar TicTacToe.class audio images
```

```
adding: TicTacToe.class (in=3825) (out=2222) (deflated 41%)
        adding: audio/ (in=0) (out=0) (stored 0%)
        adding: audio/beep.au (in=4032) (out=3572) (deflated 11%)
        adding: audio/ding.au (in=2566) (out=2055) (deflated 19%)
        adding: audio/return.au (in=6558) (out=4401) (deflated 32%)
        adding: audio/yahoo1.au (in=7834) (out=6985) (deflated 10%)
        adding: audio/yahoo2.au (in=7463) (out=4607) (deflated 38%)
        adding: images/ (in=0) (out=0) (stored 0%)
        adding: images/cross.gif (in=157) (out=160) (deflated -1%)
        adding: images/not.gif (in=158) (out=161) (deflated -1%)
```

JAR file TicTacToe.jar is compressed.

The Jar tool **compresses** files by **default**. You can turn off the compression feature by using the **0** (zero) option, so that the command would look like:

Let's use the Jar tool to **list the contents** of the TicTacToe.jar file we created in the previous section:

```
jar tf TicTacToe.jar
```

This command displays the contents of the JAR file to stdout:

```
META-INF/MANIFEST.MF
TicTacToe.class
audio/
audio/beep.au
audio/ding.au
audio/return.au
audio/yahoo1.au
audio/yahoo2.au
images/
images/cross.gif
images/not.gif
```

The JAR tool will **display additional information** if you use the v option:

```
jar tvf TicTacToe.jar
```

For example, the **verbose** output for the TicTacToe JAR file would look similar to this:

```
 256 Mon Apr 20 10:50:28 PDT 1998 META-INF/MANIFEST.MF
3885 Mon Apr 20 10:49:50 PDT 1998 TicTacToe.class
   0 Wed Apr 15 16:39:32 PDT 1998 audio/
4032 Wed Apr 15 16:39:32 PDT 1998 audio/beep.au
2566 Wed Apr 15 16:39:32 PDT 1998 audio/ding.au
6558 Wed Apr 15 16:39:32 PDT 1998 audio/return.au
7834 Wed Apr 15 16:39:32 PDT 1998 audio/yahoo1.au
7463 Wed Apr 15 16:39:32 PDT 1998 audio/yahoo2.au
   0 Wed Apr 15 16:39:44 PDT 1998 images/
 157 Wed Apr 15 16:39:44 PDT 1998 images/cross.gif
 158 Wed Apr 15 16:39:44 PDT 1998 images/not.gif
```

Recall that the contents of TicTacToe.jar are:

```
META-INF/MANIFEST.MF
TicTacToe.class
audio/
audio/beep.au
audio/ding.au
audio/return.au
audio/yahoo1.au
audio/yahoo2.au
images/
images/cross.gif
images/not.gif
```

Suppose you want to **extract the TicTacToe class file and the cross.gif image** file the:

```
jar xf TicTacToe.jar TicTacToe.class images/cross.gif
```

Recall that TicTacToe.jar has these contents:

```
META-INF/MANIFEST.MF
TicTacToe.class
audio/
audio/beep.au
audio/ding.au
audio/return.au
audio/yahoo1.au
audio/yahoo2.au
images/
images/cross.gif
images/not.gif
```

Suppose that you want to **add** the file images/new.gif to the JAR file:

```
jar uf TicTacToe.jar images/new.gif
```

The revised JAR file would have this table of contents:

```
META-INF/MANIFEST.MF
TicTacToe.class
audio/
audio/beep.au
audio/ding.au
audio/return.au
audio/yahoo1.au
audio/yahoo2.au
images/
images/cross.gif
images/not.gif
images/new.gif
```

Manifest File

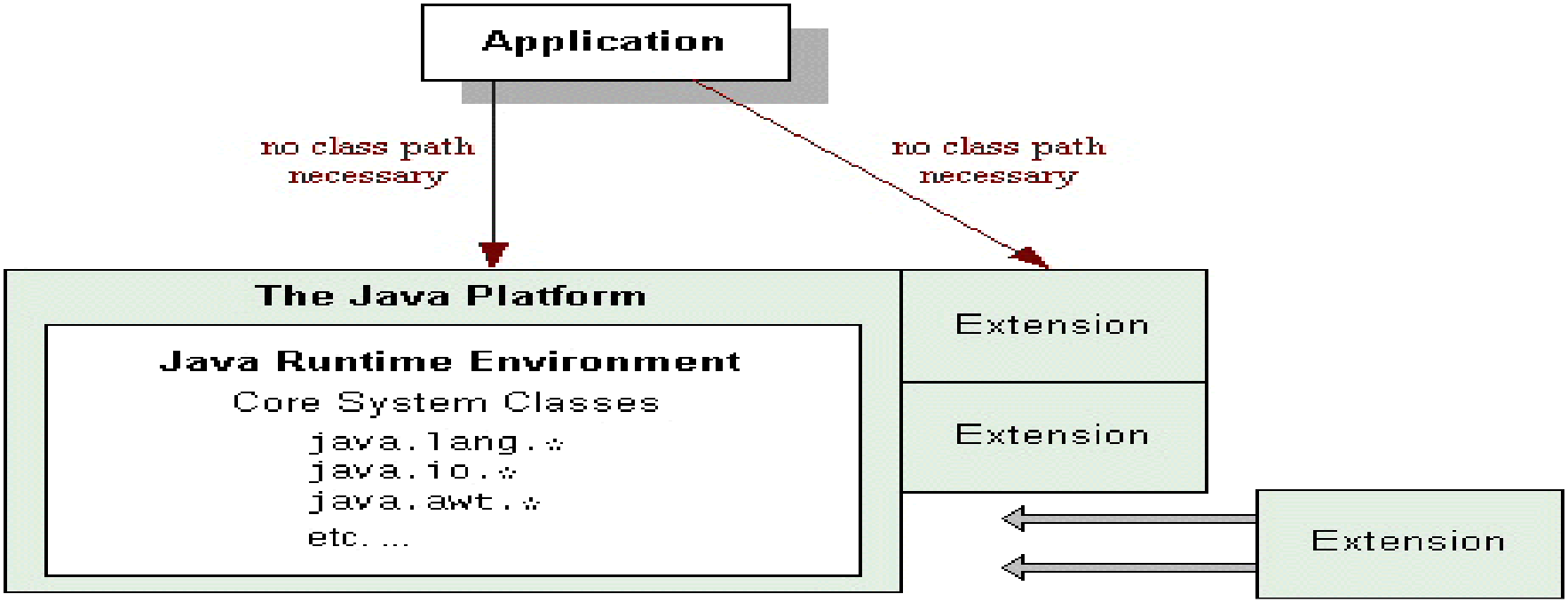
- Di dalam file JAR atau didalam direktori hasil ekstrak dari file JAR maka kita akan menemukan file **MANIFEST.MF** di dalam direktori META-INF.
- File MANIFEST.MF adalah metafile yang menyediakan berbagai informasi dalam file JAR.
- Secara default isi file MANIFEST.MF adalah:

```
Manifest-Version: 1.0
Created-By: 1.3.0 (Sun Microsystem Inc.)
```

Header Main-Class

- Header Main-Class digunakan agar Java mengetahui file main yang digunakan untuk mengeksekusi program Java.
- Cara membuat header Main-Class
  - Sintaks : Main-Class: <namafilemain>
  - Contoh : Main-Class: kelasMain
- Cara menambah (mengupdate) file MANIFEST.MF, yang disimpan pada file MANIFEST.MF, file jar bernama CobaUniv.jar

jar cmf MANIFEST.MF univ.jar id CobaUniv.class



Java Extensions

Extensions are "add-on" modules to the Java platform. Their classes and public APIs are automatically available to any applications running on the platform.

Creating and Using Extensions

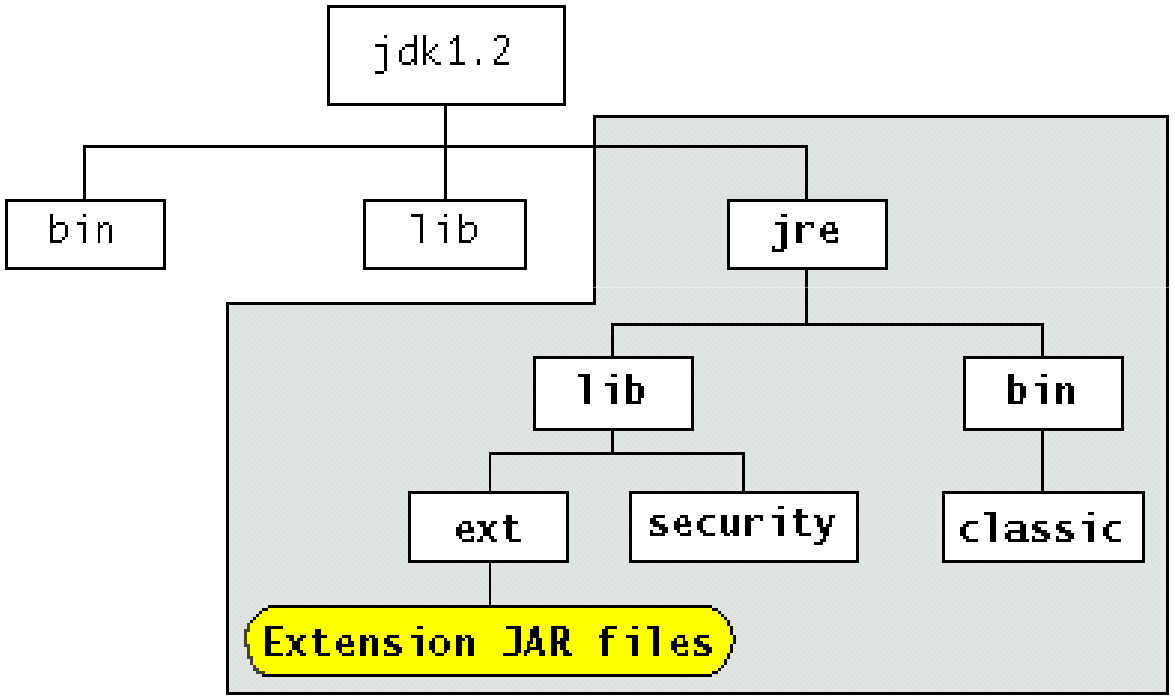
There are two ways of using extensions:

•by placing the JAR file in a special location in the directory structure of the Java Runtime Environment, in which case it's called an **installed extension**.

•by referencing the JAR file in a specified way from the manifest of the another JAR file, in which case it's called a **download extension**.

Installed Extensions

Installed extensions are JAR files in the **lib/ext** directory of the Java Runtime Environment (JRE) software.



any JAR file in the JRE's lib/ext directory will be automatically treated by the runtime environment as an extension.

Downloaded Extensions

Set Classpath to Downloaded Extensions

Class-Path: b.jar

Then the classes in b.jar serve as extension classes for purposes of the classes in a.jar.

If b.jar weren't in the same directory as a.jar, then the value of the Class-Path header should be set to **the relative pathname of b.jar**.

JAR Hell

- Jika file JAR sebenarnya sama namun beda versi, Java tidak bisa membedakannya, sehingga kedua file JAR tersebut tetap dapat diletakkan pada folder Extensions yang sama
- Hal ini menyebabkan kemungkinan terjadinya JAR HELL

JDBC

- Java Database Connectivity?
- Java menyediakan JDBC yang berfungsi untuk berhubungan dengan database.
- Database yang didukung oleh Java cukup banyak, seperti : MySQL, Postgres, Oracle, DB2, Access dan lain-lain.
- JDBC berisi kumpulan kelas-kelas dan interface yang ditulis dengan bahasa Java.

JDBC (2)

- Yang dilakukan JDBC
  - Membangun koneksi ke data source
  - Mengirim statement ke data source
  - Memproses hasil statement tersebut

- Java menyediakan tiga produk JDBC:
  - JDBC driver manager
  - JDBC driver test suite
  - JDBC ODBC bridge

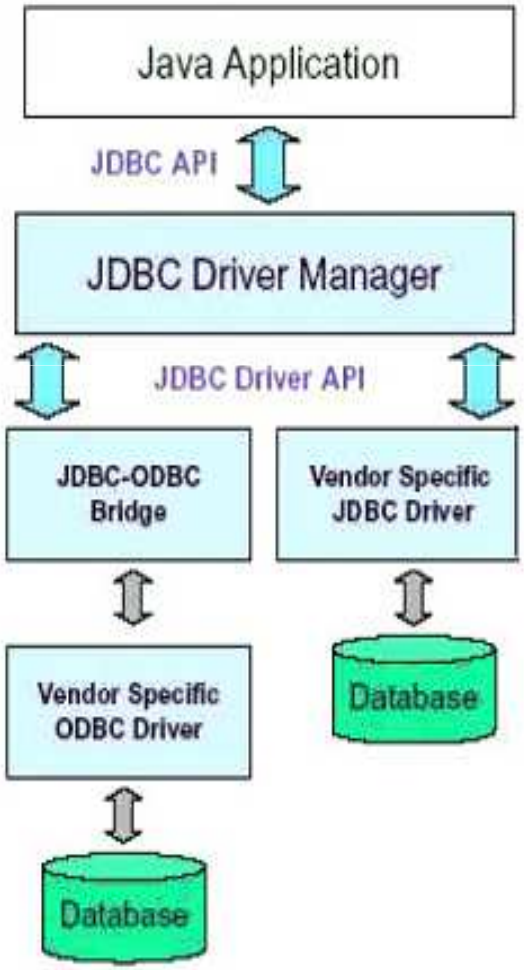
ODBC vs JDBC

- ODBC tidak cocok dipakai langsung dengan Java karena ditulis dengan bahasa C, pemanggilan dari Java ke C memiliki masalah keamanan, implementasi, robustness, dan portabilitas sistem.
- Penerjemahan dari C ke Java tidak akan berhasil baik. Contoh: Java tidak memiliki pointer.
- ODBC sulit dipelajari karena optionnya yang sulit walaupun untuk query yang sederhana.
- Java API diperlukan untuk mempertahankan solusi “murni Java”, agar dapat berjalan di berbagai platform. Karena ODBC harus diinstall dahulu di setiap client dan tidak semua platform.

Keunggulan JDBC

- Mempertahankan data enterprise yang ada
- Menyederhanakan development enterprise
- Tidak memerlukan konfigurasi pada jaringan komputer
- Akses penuh ke meta data
- Koneksi database menggunakan URL dan DataSource (yang meyediakan connection pooling dan distributed transaction)

Arsitektur JDBC



Lapisan Vendor Specific JDBC Driver merupakan driver JDBC yang dikeluarkan oleh para vendor pengembang RDBMS.

Sedangkan JDBC- ODBC Bridge berfungsi sebagai perantara untuk mengakses database melalui ODBC driver.

Baik JDBC driver maupun JDBC-ODBC Bridge diatur dan dapat diakses melalui JDBC Driver Manager.

Aplikasi yang kita kembangkan untuk mengakses database dengan memanfaatkan JDBC akan berinteraksi dengan JDBC Driver Manager.

JDBC API

- Tersedia dalam paket java.sql dan javax.sql.
- **DriverManager** – memanggil driver JDBC ke memori, dan dapat juga digunakan untuk membuka koneksi ke sumber data.
- **Connection** – mempresentasikan suatu koneksi dengan suatu data source, juga digunakan untuk membuat objek Statement, PreparedStatement dan CallableStatement.
- **Statement** – mempresentasikan suatu perintah SQL, dan dapat digunakan untuk menerima objek ResultSet.

JDBC API (2)

- **PreparedStatement** – merupakan alternatif untuk objek Statement SQL yang telah terkompilasi awal.
- **CallableStatement** – mempresentasikan suatu stored procedure, dan dapat digunakan untuk menjalankan stored procedures yang terkompilasi dalam suatu RDBMS yang mendukung fasilitas tersebut.
- **ResultSet** – mempresentasikan sebuah hasil dari database yang dihasilkan dari statemen SQL SELECT.
- **SQLException** – suatu class exception yang membungkus kesalahan (error) pengaksesan database.

JDBC Data Type

JDBC Type	Java Type
BIT	boolean
TINYINT	byte
SMALLINT	short
INTEGER	int
BIGINT	long
REAL	float
FLOAT DOUBLE	double
BINARY VARBINARY LONGVARBINARY	byte[]
CHAR VARCHAR LONGVARCHAR	String

JDBC Type	Java Type
NUMERIC DECIMAL	BigDecimal
DATE	java.sql.Date
TIME TIMESTAMP	java.sql.Timestamp
CLOB	Clob*
BLOB	Blob*
ARRAY	Array*
DISTINCT	mapping of underlying type
STRUCT	Struct*
REF	Ref*
JAVA_OBJECT	underlying Java class

*SQL3 data type supported in JDBC 2.0

Pemrograman JDBC

- Membangun koneksi
  - Memuat driver ODBC

    ```
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    ```
  - Atau

    ```
    DriverManager.registerDriver(new sun.jdbc.odbc.JdbcOdbcDriver ());
    ```

- Membangun koneksi URL
  Format: `jdbc:odbc:<nama_db>`
- Contoh lengkap:

```
String url = “jdbc:odbc:Buku”;
String user = “”;
String pass = “”;
Connection con =
DriverManager.getConnection(url,user,pass);
```

Pemrograman JDBC (2)

- Membuat Statement

  Menggunakan Obyek Connection yang sudah kita buat sebelumnya:
  - `Statement stmt = con.createStatement();`
- Menjalankan Statement
- Method **executeUpdate** untuk DDL dan DML insert, update, dan delete.
  - `String query = “delete from tabel where id=1”:`
  - `Statement stmt = con.createStatement();`
  - `int hsl = Stmt.executeUpdate(query);`
- Method **executeQuery** untuk DML select
  - `String query = “select * from tabel”;`
  - `Statement stmt = con.createStatement();`
  - `ResultSet rs = Stmt.executeQuery(query);`

Pemrograman JDBC (3)

- Mengambil hasil Statement dari Query dan Memprosesnya
- DDL dan DML: update, insert, dan delete

```
int hsl = Stmt.executeUpdate(query);
if(hsl == 1)
   System.out.println(“Berhasil”);
else
   System.out.println(“Gagal”);
```

- DML: select

```
ResultSet rs = Stmt.executeQuery(query);
while(rs.next()){
   int a = rs.getInt(“fieldA”);
   String b = rs.getString(“fieldB”);
   float c = rs.getFloat(“fieldC”);
}
```

Pemrograman JDBC (4)

- Tutup koneksi yang sudah dibuat.

```
con.close();
```

- Kita dapat membuat class yang berisi semua method yang membantu kita untuk melakukan koneksi dan transaksi ke database!

Penting!

- Harus mengetahui dan memiliki JDBC driver sesuai dengan database yang digunakan.
- Harus mengetahui cara koneksi dengan database.
- Harus mengimport `java.sql.*;`

Contoh: MySQL Create Table

```
import java.sql.*;
public class CreateTableSepatuApp {
    public static void main(String[] args) {
        String createTableSepatu =
            "CREATE TABLE SEPATU "
                + "(NAMA_SEPATU VARCHAR(40),"
                + "SUP_ID INTEGER,"
                + "HARGA REAL,"
                + "PENJUALAN INTEGER,"
                + "TOTAL INTEGER)";
        try {
            Class.forName("org.gjt.mm.mysql.Driver");
            Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sepatudb?user=root&password=");
            Statement stmt = con.createStatement();
            stmt.executeUpdate(createTableSepatu);
            System.out.println("Tabel SEPATU berhasil dibuat.");
            stmt.close();
            con.close();
        } catch (ClassNotFoundException e) {
            System.out.println("Eksepsi: " + e.getMessage());
        } catch (SQLException e) {
            System.out.println("Eksepsi SQL: " + e.getMessage());
        }
    }
}
```

Membaca Isi Data ODBC

```
public class cobaDB {
    public static void main(String[] args) {
        String selectMhs = "select * from mhs";
        try {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            String url = "jdbc:odbc:mhs";
            String user = "";
            String pass = "";
            Connection con = DriverManager.getConnection(url,user,pass);

            Statement stmt = con.createStatement();
            ResultSet rs = stmt.executeQuery(selectMhs);
            while(rs.next()){
                String nim = rs.getString("nim");
                String nama = rs.getString("nama");
                        int ipk = rs.getInt("ipk");
                System.out.println(nim + "\t" + nama + "\t" + ipk);
            }
            rs.close();
            stmt.close();
            con.close();
        } catch (ClassNotFoundException e) {
            System.out.println("Eksepsi: " + e.getMessage());
        } catch (SQLException e) {
            System.out.println("Eksepsi SQL: " + e.getMessage());
        }
    }
}
```

PreparedStatement

```
String insertBuku = "insert into buku(KodeBuku, Judul, Penerbit, ThTerbit) values (?, ?, ?, ?)";
PreparedStatement stmt = con.prepareStatement(insertBuku);
stmt.setString(1, "IPA20");
stmt.setString(2, "aaa");
stmt.setString(3, "bbb");
stmt.setString(4, "1900");
int h = stmt.executeUpdate();

if(h==1)
    System.out.println("Berhasil");
else
    System.out.println("Gagal");
```

Cursor ResultSet

- Method pergerakan kursor yang didukung oleh ResultSet:
  - previous() ke record sebelumnya
  - next() ke record selanjutnya
  - first() ke record pertama
  - last() ke record terakhir
  - absolute() ke nomor baris tertentu
  - relative() ke nomor baris dari baris sekarang
  - beforeFirst() ke nomor baris sebelum pertama
  - afterLast() ke nomor baris setelah terakhir

Cursor ResultSet

- Jika suatu ResultSet dibuat, selalu ResultSet tersebut berada pada poisi record sebelum record pertama (rs.beforeFirst()).
- Sehingga untuk mengambil data yang hanya terdiri dari satu baris, harus terlebih dahulu digunakan method rs.next() sekali.

Cursor ResultSet & Limit

- Method untuk mengambil jumlah baris:
  - getRow() yang mengembalikan nilai integer
- Method untuk membatasi jumlah baris hasil query select:
  - Statement.setFetchSize(number)

setFetchSize()

- setFetchSize() memiliki arah, yaitu:
  - ResultSet.FETCH_FORWARD untuk proses maju
  - ResultSet.FETCH_REVERSE untuk proses berbalik
  - ResultSet.FETCH_UNKNOWN untuk proses yang tidak diketahui
- Contoh:

```
Statement stmt = con.createStatement();
stmt.setFetchDirection(ResultSet.FETCH_FORWARD);
stmt.setFetchSize(30);
ResultSet rs = stmt.executeQuery(…);
```

Kembalian ResultSet

- null
  - Untuk metode getXXX yang mengembalikan obyek
- 0
  - Untuk metode getXXX yang mengembalikan tipe data primitif biasa
- false
  - Untuk metode getXXX yang mengembalikan tipe data boolean.

Exception dalam JDBC

- SQLException: ketika ada masalah pengaksesan data
- SQLWarning: ketika ada peringatan
- DataTruncation: ketika data mungkin terpotong
- BatchUpdateException: ketika tidak semua perintah update berhasil dilakukan.

The End

- TAS: open books
- Soal: pilihan ganda dan essay!