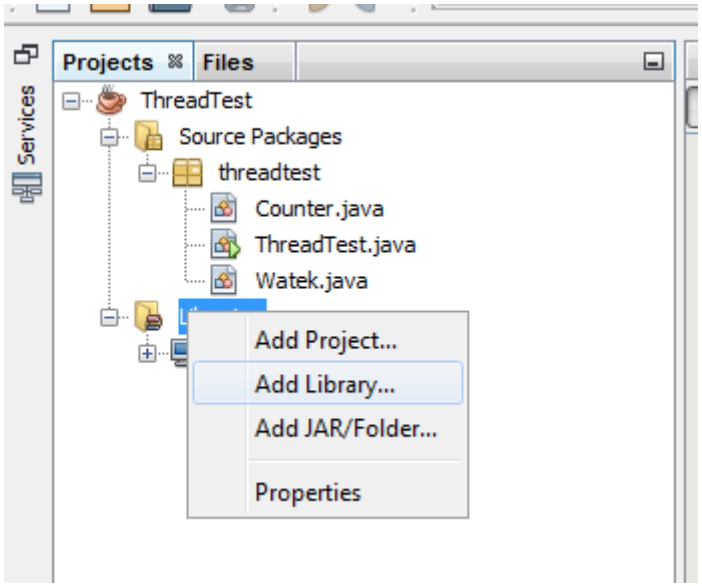
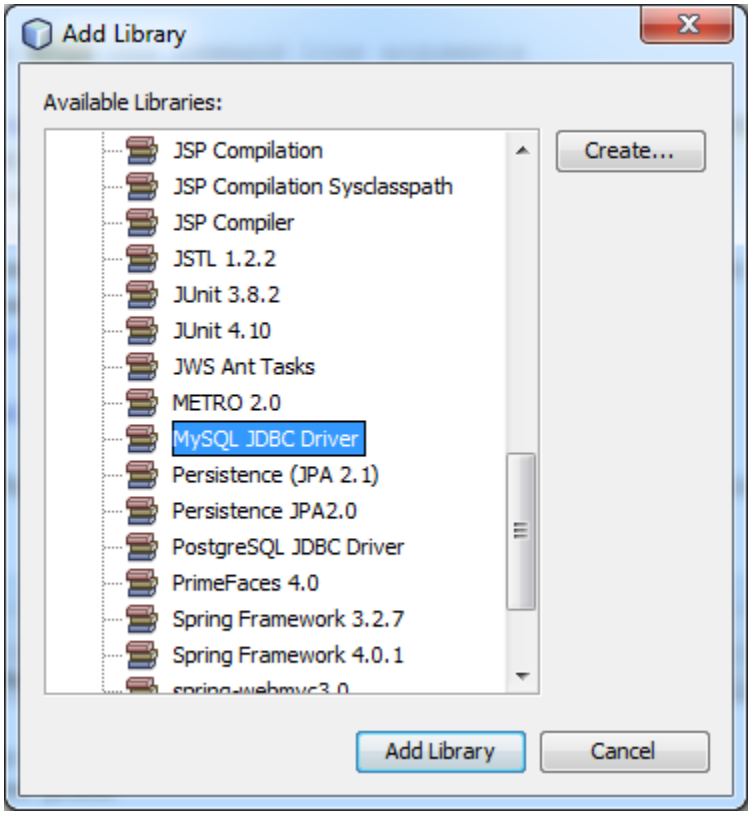
Wykorzystanie bazy danych

Korzystamy z mySQL

- NetBeans ma już wbudowany konektor jConnector MySQL:





jConnector

- Problem tylko w tym, że wbudowany kontroler jest trochę stary. Ma problemy z wyciekami pamięci. Przy zastosowaniach produkcyjnych należy zaopatrzyć się w nową wersję prosto z MySQL

Połączenie z bazą danych

```
            String JDBC_DRIVER = "com.mysql.jdbc.Driver";
            String DB_URL = "jdbc:mysql://155.158.112.142/tomek?dontTrackOpenResources=true";
            String USER="costam";
            String PASS="costam";
        //załadowanie i inicjalizacja sterownika JDBC
        Class.forName(JDBC_DRIVER);
        //właściwe połączenie
        Connection con = DriverManager.getConnection(DB_URL,USER, PASS);
        }
        catch(ClassNotFoundException e) {
            System.out.println("Błąd sterownika JDBC" + e.getMessage());
        }

        catch(SQLException e) {
            System.out.println("Błąd SQLa " + e.getMessage());
        }
```

ThreadTest > main > try > catch SQLException e

Output - ThreadTest (run)

```
run:
Błąd SQLaAccess denied for user 'costam'@'[redacted]' (using password: YES)
BUILD SUCCESSFUL (total time: 0 seconds)
```

Odczytujemy informacje z bazy danych

```
Connection con = DriverManager.getConnection(DB_URL,USER, PASS);
String query = "SELECT * from test";
    try {
        PreparedStatement p = con.prepareStatement(query);
        ResultSet result = p.executeQuery();

        while(result.next()) {
        System.out.println(result.getInt(1));
        }

    } catch (SQLException e) {
        System.out.println("Blad w query: " + e.getMessage());
    }
}
```

Kilka uwag - ResultSet

- Klasa ResultSet pozwala nam na iterowanie po wynikach zapytania typu SELECT
  - Metoda first() przenosi nas ze wskaźnikiem odczytu na początek zbioru wynikowego. Jeśli SELECT zwrócił nic, wtedy first() zwraca null
  - Metoda next() zwraca wartość prawda/fałsz i pozwala na przechodzenie po wszystkich krotkach wynikowych

Kilka uwag - ResultSet

- Aby otrzymać konkretną wartość musimy zdefiniować o którą kolumnę nam chodzi oraz jakiego typu są dane:
  - Metody getInt(), getString(), getFloat(), itp. przyjmują dwie postacie:
    - getInt(nr kolumny) – **NUMERY KOLUMN SĄ INDEKSOWANE OD 1**
    - getInt(String nazwa_kolumny)

Manipulujemy bazą danych

```
try(PreparedStatement p = con. prepareStatement(query)) {
                    int res = p.executeUpdate();

} catch (SQLException e) {
    System.out.println("Blad w query: " + e.getMessage());
}
```

res – zwraca informację ile wierszy zostało zmienionych przez zapytanie

Wycieki pamięci

- Pomimo istnienia Garbage Collectora, w Javie też istnieją wycieki pamięci. Najłatwiej je zauważyć właśnie przy pracy z bazami danych. Spróbuj wykonać dzisiejszy kod w pętli 1000 razy i zobacz jak rośnie wielkość zużytej pamięci.
- Aby sobie z tym radzić można wykorzystać mechanizm try-with-resources (od Javy 1.7)

try-with-resources

```
Connection con = DriverManager.getConnection(DB_URL,USER, PASS);
String query = "SELECT * from test";
    try(PreparedStatement p = con.prepareStatement(query)) {
        try(ResultSet result = p.executeQuery()) {
            while(result.next()) {
                System.out.println(result.getInt(1));
            }
        }
    } catch (SQLException e) {
        System.out.println("Blad w query: " + e.getMessage());
    }
```

Odwołanie do zasobów (PreperedStatement i ResultSet) **wewnątrz** try().

Dzięki temu zasoby zostaną automatycznie zwolnione po ich wykorzystaniu.

Ćwiczenia

1. Połączenie z bazą danych zwykle realizujemy za pomocą Singletonu. Napisz taką klasę korzystającą z tego wzorca projektowego do połączenia z bazą danych. Oprogramuj do tego metody pozwalające na wygodne korzystanie z bazy, np.:
    1. ResultSet executeQuery(String query)
    2. Int executeUpdate(String query)
2. Weź swoją książkę adresową z poprzednich zajęć i dołóż do niej zapis i odczyt z bazy danych.
3. Porównaj które operacje są szybsze: korzystające z silnika bazodanowego, czy lokalne korzystające z Javy. W tym celu stwórz tablicę o rozmiarze kilku milionów zawierającą pola ID(BIGINT) oraz DATA(BIGINT). Sprawdź jak długo trwa posortowanie wartości za pomocą polecenia SQLowego i za pomocą Javy.