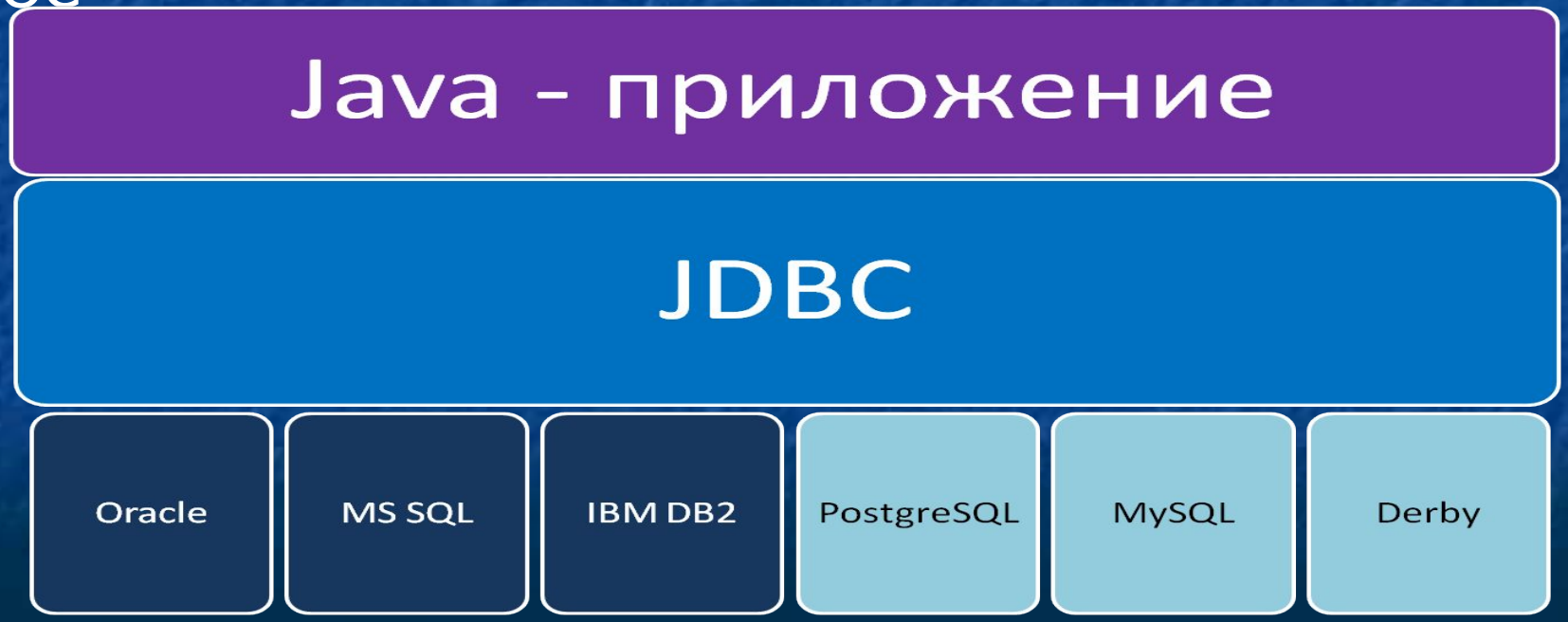
Основы JDBC

Особенности JDBC

- JDBC (Java DataBase Connectivity) – интерфейс, при помощи которого Java-приложения взаимодействуют с базами данных и манипулируют с их данными
- Драйвер JDBC реализует интерфейс с конкретной БД.
- Основное преимущество – возможность выполнения одного и того же программного кода для различных СУБД и ОС



Типы JDBC-драйверов

1. Драйвер транслирует JDBC в ODBC и для взаимодействия с БД используется драйвер ODBC (Microsoft Open DataBase Connectivity). В состав JDK включен драйвер – Мост JDBC/ODBC. Не удобен, поскольку требует установки и конфигурации. Для тестирования
2. Пишется частично на Java, частично на собственном языке. Необходимо помимо библиотеки Java установить платформа зависимый код
3. Создается только на основе Java с использованием независимого от БД протокола взаимодействия сервера и БД
4. Основывается на библиотеке Java, транслирующей JDBC-запросы в протокол конкретной БД

Наиболее предпочтительны JDBC-драйверы типа 3 и 4

Основы программирования интерфейса JDBC

- Добавление драйвера осуществляется через указание пути с помощью аргумента classpath

  java –classpath c:\OracleDriver.zip

  java –classpath postgersql.jar

  java –classpath oracledriver.jar

  или изменение параметра CLASSPATH среды, или копированием драйвера в jre/lib/ext или свой проект
- URL-указатель базы данных имеет следующий формат (источник БД и его параметры)

  jdbc:название_протокола:другие_сведения

пример

  jdbc:oracle:thin:@control_mipt:1521:ORA

Подключение драйвера

- За работу с JDBC-драйвером отвечает java.sql.DriverManager. Сделать драйвер видимым для него можно несколькими способами
- Через командную строку
  java –Djdbc.drivers=oracle.jdbc.driver.OracleDriver
- Считать из property-файла
  jdbc.drivers=oracle.jdbc.driver.OracleDriver
  System.setProperty("jdbc.drivers", oracle.jdbc.driver.OracleDriver")
- Для задания нескольких драйверов используется : (двоеточие)
  oracle.jdbc.driver.OracleDriver:COM.cloudscape.core.JDBCDriver
- Регистрация вручную путем загрузки класса
  Class.forName("oracle.jdbc.driver.OracleDriver")
  Получение ссылки на драйвер java.sql.Driver
  Driver d = (Driver)Class.forName("sun.jdbc.odbc.JdbcOdbcDriver").newInstance();

Получение соединения и отправка запросов

- Соединением между программой и БД управляет объект, реализующий интерфейс java.sql.Connection. Ссылку на объект этого класса можно получить через DriverManager

```
Connection connection = DriverManager.getConnection(url, user, password)
```

- Объекты этого класса дают возможность программам создавать запросы SQL. Отправка запроса осуществляется через объекты класса java.sql.Statement

```
Statement statement =
connection.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY)
```

- Отправка SQL запросов осуществляется через вызов метода executeQuery(String query)

```
ResultSet resultSet = statement.executeQuery("SELECT * FROM TestTable")
```

- Соединение с БД закрывается при помощи метода close()

```
connection.close()
```

Однако следует помнить, что все транзакции должны быть завершены и при необходимости зафиксированы

- Фиксация и откат транзакций осуществляется при помощи методов commit(), setAutoCommit() и rollback()

Обработка результатов запроса

- Результаты запроса упаковываются в объект класса java.sql.ResultSet. Класс содержит большой набор методов работы с данными
- Получение метаданных осуществляется через вызов метода getMetaData() и объекты класса java.sql.ResultSetMetaData

```
ResultSetMetaData metaData = resultSet. getMetaData()
```

- Объекты класса ResultSetMetaData в частности содержат информацию об атрибутах сформированной сущности. Например, количество атрибутов - getColumnCount(), их типы – getColumnType(int column) (см. класс java.sql.Types), имена - getColumnName(int column)
  - Нумерация элементов всех типов начинается с 1
- Значения данных из текущей строки таблицы для конкретного атрибута извлекаются при помощи методов типа getObject(int column) (getInt() и т.п.)

```
Object element =  resultSet.getObject(1); //Значение в первом атрибуте.
```

- Перемещение по строкам осуществляется различными методами, основной – boolean next() – последовательное перемещение.

```
while (resultSet.next()) {
      for (int i = 1; i < metaData.getColumnCount(); i++)
            Object element =  resultSet.getObject(1)
}
```

absolute(int row) – переход к конкретной строке