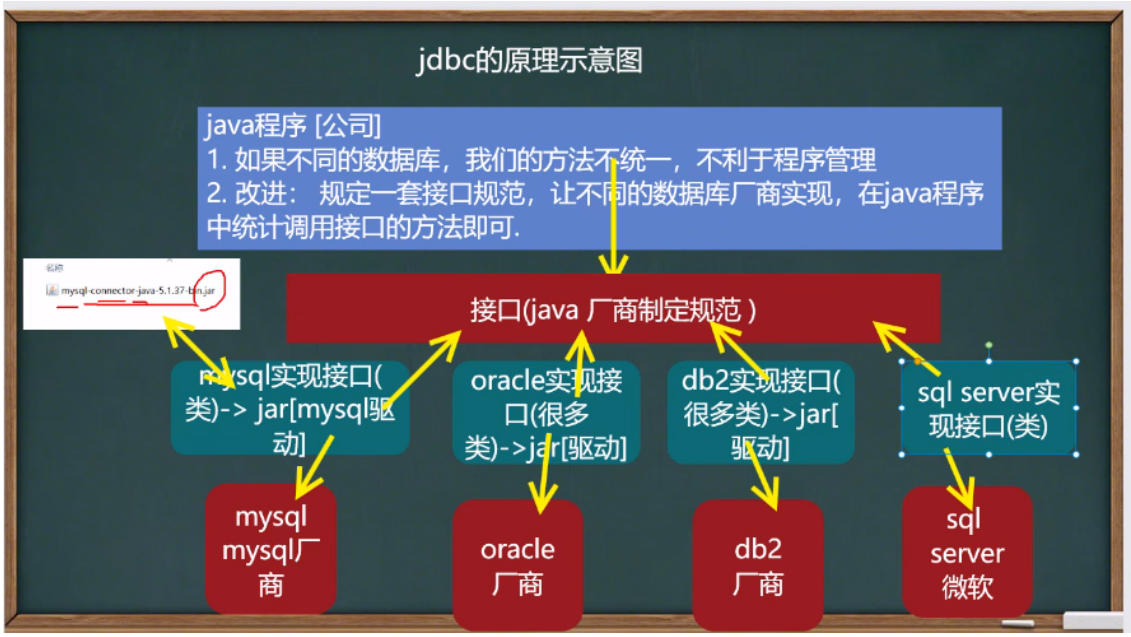
JDBC与数据库连接池

JDBC概述

1. JDBC为访问不同的数据库提供了统一的接口，为使用者屏蔽了细节问题。
2. Java程序员使用JDBC，可以链接任何提供了JDBC驱动程序的数据库系统，从而完成对数据库的各种操作。
3. JDBC的基本原理：



4. 模拟JDBC

JDBC接口

```
package com.diduox.jdbc.myjdbc;

public interface jdbcinterface {

    //链接
    public Object getConnection();
    //crud
    public void crud();
    //关闭链接
    public void close();

}
```

mysql驱动

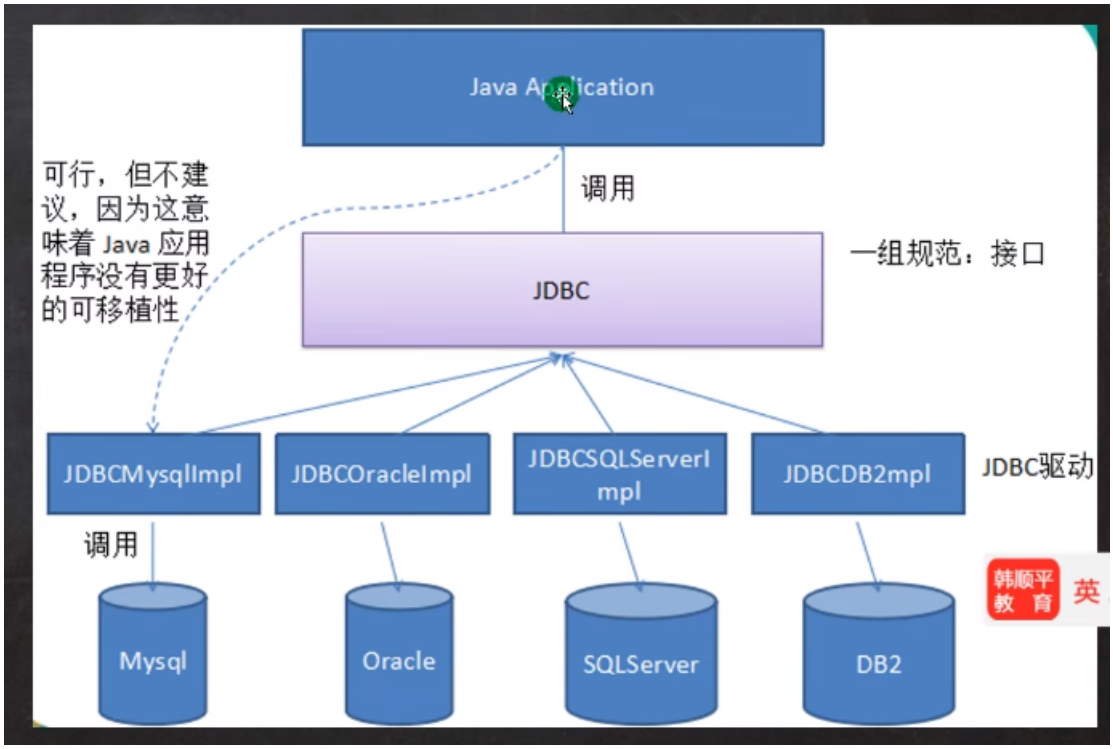
```
package com.diduox.jdbc.myjdbc;

public class MYsqlJdbcImpl implements jdbcinterface{
    @Override
    public Object getConnection() {
        System.out.println("得到mysql的链接");
        return null;
    }
```

```
    @Override
    public void crud() {
        System.out.println("完成Mysql的增删改查");
    }

    @Override
    public void close() {
        System.out.println("关闭mysql的链接");
    }
}
```

实现

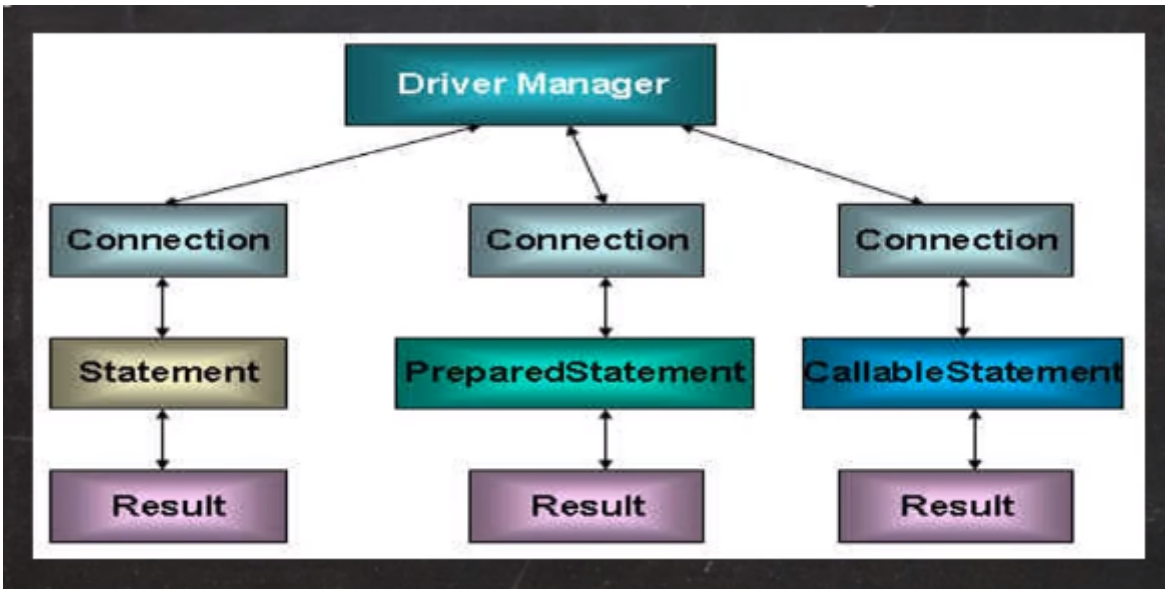
```
package com.diduox.jdbc.myjdbc;

public class TestJDBC {
    public static void main(String[] args) {
        //完成对mysql的操作
        jdbcinterface jdbcinterface = new MYsqlJdbcImpl();
        jdbcinterface.getConnection();//通过接口来调用实现类
        jdbcinterface.crud();
        jdbcinterface.close();
    }
}
```

JDBC带来的好处



JDBCAPI

JDBC API是一系列接口，它统一和规范了应用程序和数据库的链接、执行SQL语句，并得到返回结果等各类操作。



JDBC快速入门

JDBC程序编写步骤

1. 注册驱动-加载Driver 类
2. 获取链接-得到Connection类
3. 执行增删改查-发送SQL 给mysql执行
4. 释放资源-关闭相关链接

例：通过JDBC对 表 ACTOR进行添加，删除，和修改操作。

```
USE db02;
CREATE TABLE actor(
    id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(32) NOT NULL DEFAULT '',
    sex char(1) NOT NULL DEFAULT '女',
    borndate datetime,
    phone VARCHAR(12)
);
```

```
package com.diduox.jdbc;

import com.mysql.jdbc.Driver;
import java.sql.Connection;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Properties;

/*
* 这是第一个Jdbc程序，完成简单的操作
* */
public class Jdbc01 {
    public static void main(String[] args) throws SQLException {
    //前置工作：在项目下创建一个文件夹 比如libs
```

```
    //将 mysql.jar 拷贝到该目录下，点击add to project .. 加入到该项目

    //1.注册驱动
    Driver driver = new Driver();//创建driver链接
    //2.得到链接 通过jdbc的方式连接mysql
    String url = "jdbc:mysql://localhost:3306/db02";
    //将用户名和密码放入到Properties对象
        Properties properties = new Properties();
        properties.setProperty("user","root");//用户
        properties.setProperty("password","");//密码

        Connection connect = driver.connect(url, properties);
    //3.执行sql
        String sql = "INSERT INTO actor VALUES(null,'王向晚','女','1970-11-
11','110')";
        //执行静态的sql语句，并返回执行的结果
        Statement statement = connect.createStatement();
        int i = statement.executeUpdate(sql);//如果是dml语句，返回的就是影响的行数

        System.out.println(i > 0 ? "成功" : "失败");
    //4.关闭链接资源
        statement.close();
        connect.close();
    }
}
```

连接数据库的5种方式

方式1：使用Driver实现类对象

属于静态加载，会直接使用com.mysql.jdbc.Driver(),属于静态加载，灵活性差，依赖性强。

```
package com.diduox.jdbc;

import com.mysql.jdbc.Driver;
import java.sql.Connection;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Properties;

public class JdbcConn1 {
    public static void main(String[] args) throws SQLException {
        Driver driver = new Driver();
        String url = "jdbc:mysql://localhost:3306/db02";
        Properties properties = new Properties();
        properties.setProperty("user","root");
        properties.setProperty("password","");
        Connection connect = driver.connect(url, properties);
        String sql = "INSERT INTO actor VALUES(null,'王嘉然','女','1971-12-
12','120')";
        Statement statement = connect.createStatement();
        int rows = statement.executeUpdate(sql);
        statement.close();
        connect.close();
```

```
    }
}
```

方式2：使用反射加载

方式3：使用DriverManage 替代 Driver 进行统一管理

方式4：使用Class.forName自动完成注册驱动，简化代码

方式5：在方式4的基础上改进，使用配置文件，链接数据库更加灵活