**一．JDBC**

JDBC:java应用程序和数据库【多种】之间的连接桥梁 JAVA DATE BASE Connectivity

1.加载驱动，不同数据库，参数不同

Class.*forName*("oracle.jdbc.driver.OracleDriver");

2.连接数据库

Connection conn = DriverManager.*getConnection*("jdbc:oracle:thin:@localhost:1521:orcl","CLEAR","sql");

3.创建SQL语句

String sql="";

1. 执行SQL语句，创建Statement对象，调用execute等方法

Statement state = conn.createStatement();

ResultSet rs = state.executeQuery(sql);

Date日期的类型转换：

java.util.Date是java中的数据类型

通过statement或preparestatement将java应用程序中的数据类型【java.util.Date】

传入sql语句时需要进行强制类型转换 new Date 导入java.sql.Date包。

反之不需要进行类型转换，因为java.sql.Date是父类

几种常用数据库连接方式：

1. JDBC+ODBC桥的方式

需要数据库的ODBC驱动，仅适用于微软的系统

1. JDBC+厂商API的方式

特点：厂商的API一般使用C编写

1. JDBC+厂商Database Connection Server+DataBase

在Java与数据库之间架起一台专门用于数据库连接的服务器，一般由数据库厂商提供

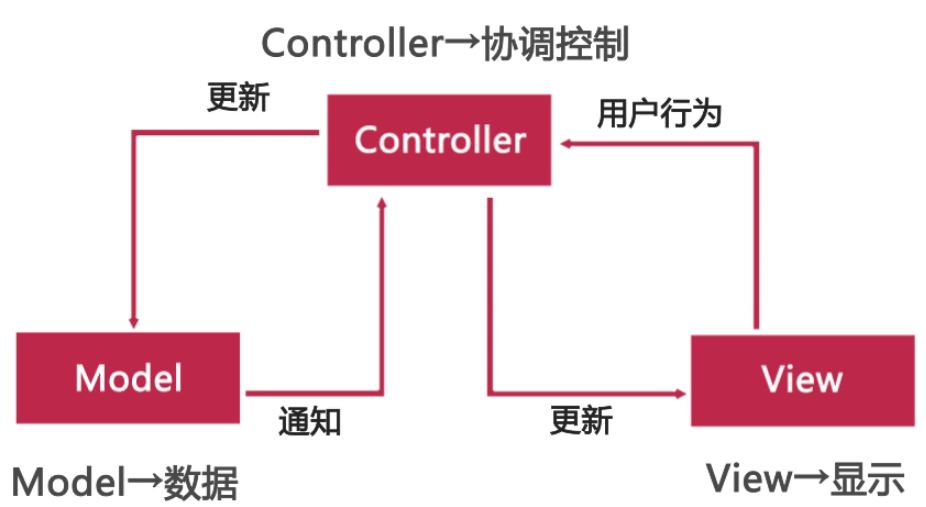
效率略快，不便捷

1. JDBD+DATABASE【当前使用】

使得Application与数据库分开，开发者只需要关心内部逻辑的实现而不需注重数据库连接的具体实现

开发中JDBC的替代工具【orm映射工具】：mybatis hibernate

**二．MVC**





View：视图层 为用户提供UI，重点关注数据的呈现

Model：模型层 业务数据的信息表示，关注支撑业务的信息构成，通常是多个业务实体的组合

Controller：控制层 调用业务逻辑产生合适的数据，传递数据给视图层用于呈现

MVC：是一种架构模式和思考方式，程序分层，分工合作，既相互独立又协同工作

**三．网络架构**

1.网络应用程序的架构

(2).Client/Server简称CS架构

B.三层的CS架构

特点：所有的业务逻辑处理都有应用服务器来做。

优点：可移植性好（一般应用服务器都是有Java语言来写的），适合大型的企业级开发

缺点：客户端需要安装和维护，开发复杂。

(3).Browser/Server简称BS架构

特点：使用浏览器来充当客户端，使用web服务器来充当服务端，使用标准的http协议来通信。

优点：开发相对简单，不需要单独安装客户端了