主要内容：

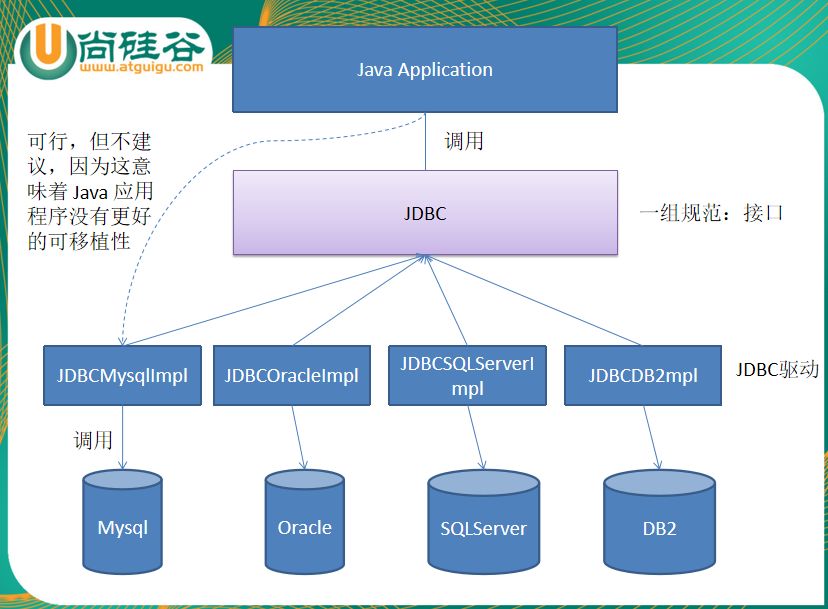
* JDBC简介
* 基本环境搭建
* Driver
* DriverManager
* 代码优化

1、JDBC简介

数据库是实现持久化的一种途径，而JDBC则是通向数据库的桥梁。

通俗地讲，JDBC就是一组API（包括少量类），为访问不同数据库提供了统一的途径，为开发者屏蔽了一些细节问题。比如，我们都知道浏览器发送HTTP请求访问服务器，但其实请求底层仍是TCP协议。同样的，访问数据库底层也通过TCP协议。你知道怎么与数据库建立TCP连接吗？一部分科班读者可能对计算机网络非常熟悉，但是大部分像我这样的野生程序员可能压根没想过这个问题。所幸，这些具体的实现，各大数据库产商已经替我们做了。

JDBC是Java制定的接口，数据库产商依照该接口编写与自家数据库配套的实现类。比如MySQL、Oracle、SqlServer等都有自己的不同实现，这些实现类的集合既是我们笼统意义上的“驱动”。



**面向接口编程**

在代码中直接new具体的驱动类，会使程序高度耦合。比如，后期如果要切换数据库（虽然很少），就要临时调换驱动类，需要修改源码，不符合开闭原则。而面向接口编程，实际上就是一种“多态”。屏蔽具体的实现，只需调用接口方法，传入规定的参数即可得到预期的返回值。切换数据库驱动并不影响程序运行结果。

2、基本环境搭建

上面说过，JDBC只是一组接口，具体实现交给驱动。所以，要使用JDBC完成CRUD，必须先导入具体的数据库驱动。

Mysql的驱动：

@Test  
public void testDriver() throws SQLException {  
 *//1. 创建一个 Driver 实现类的对象* Driver driver = new com.mysql.jdbc.Driver();  
  
 *//2. 准备连接数据库的基本信息: url, user, password* String url = "jdbc:mysql://192.168.136.128:3306/test";  
 Properties info = new Properties();  
 info.put("user", "root");  
 info.put("password", "root");  
  
 *//3. 调用 Driver 接口的　connect(url, info) 获取数据库连接* Connection connection = driver.connect(url, info);  
 System.***out***.println(connection);  
}

Oracle的驱动：

@Test  
public void testDriver() throws SQLException{  
  
 *//获取OracleDriver对象* OracleDriver oracleDriver = new OracleDriver();  
 *//准备数据库连接的基本信息* String url = "jdbc:oracle:thin:@10.142.141.16:1521:orasit02";  
 Properties properties = new Properties();  
 properties.put("user", "CCMS\_SIT\_D");  
 properties.put("password", "CCMS\_SIT");  
 *// 创建数据库的连接*

*//方式一：直接通过Driver进行数据库连接* Connection connect = oracleDriver.connect(url, properties);  
 System.***out***.println(connect);

*//方式二：  
//加载数据库驱动程序*Class.forName("oracle.jdbc.OracleDriver");  
*//通过DriverManager的getConnection()获取连接*Connection connection = DriverManager.getConnection(url, properties);  
System.***out***.println(connection);

}

直接使用Driver进行数据库连接与 使用驱动进行数据库连接有什么区别？

DriverManager是java.sql包下的一个类，内部维护着一个CopyOnWriteArrayList用于存放已经注册的驱动实例。之所以能getConnection()，是因为底层会循环遍历所有驱动，找到当前注册的驱动后调用driver.connect()获得Connection。

其实也没为啥，人家设计的初衷就是为了支持注册多个（多种）驱动，通过DriverManager可以管理多个驱动程序。所以它叫“驱动管理器”(DriverManager)。

静态代码块注册驱动到DriverManager对于Driver方式获取Connection是多余的，但是对于DriverManager获取Connection是必须的。

推荐使用DriverManager，数据库驱动连接。

使用JDBC的三大步骤：

1. 连接数据库
2. 执行sql语句
3. 获取结果

1、连接数据库：

总之，就是根据给定的url和用户名密码，返回一个与数据库关联的Connection。JDBC相当于是程序与数据库之间的桥梁。得到Connect代表桥已经建好，此刻已经可以通车了。

*//获取连接* connection = JdbcUtils\_02.getConnection();  
 *//创建模板* String sql = "select a.surname||a.name, a.agentnum, a.agentcode from laagent a where a.agentcode = ?";  
 preparedStatement = connection.prepareStatement(sql);  
 *//传参* preparedStatement.setNString(1, "BXA0000003");  
 *//获取结果集* resultSet = preparedStatement.executeQuery();  
 *//处理结果集* while(resultSet.next()){  
 System.***out***.println(resultSet.getObject(1) + ","  
 + resultSet.getObject(2) + "," + resultSet.getObject(3));  
 }  
} catch (SQLException e) {  
 e.printStackTrace();  
}finally {  
 *//释放连接资源* JdbcUtils\_02.free(resultSet,preparedStatement,connection);  
}

*//释放连接*public static void free(ResultSet rs, PreparedStatement st, Connection conn){  
 try {  
 if (rs != null){  
 rs.close();  
 }  
 }catch (SQLException e) {  
 e.printStackTrace();  
 }finally {  
 try{  
 if(st != null){  
 st.close();  
 }  
 }catch (SQLException e){  
 e.printStackTrace();  
 }finally {  
 try{  
 if(conn != null) {  
 conn.close();  
 }  
 }catch (SQLException e){  
 e.printStackTrace();  
 }  
 }  
 }  
}