day30 综合案例-学员管理系统【C/S版】

今日内容

- 学员管理系统

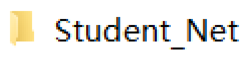
学习目标

- [ ] 能够完成客户端添加功能
- [ ] 能够完成客户端修改功能
- [ ] 能够完成客户端删除功能
- [ ] 能够完成客户端获取功能
- [ ] 能够完成服务端功能

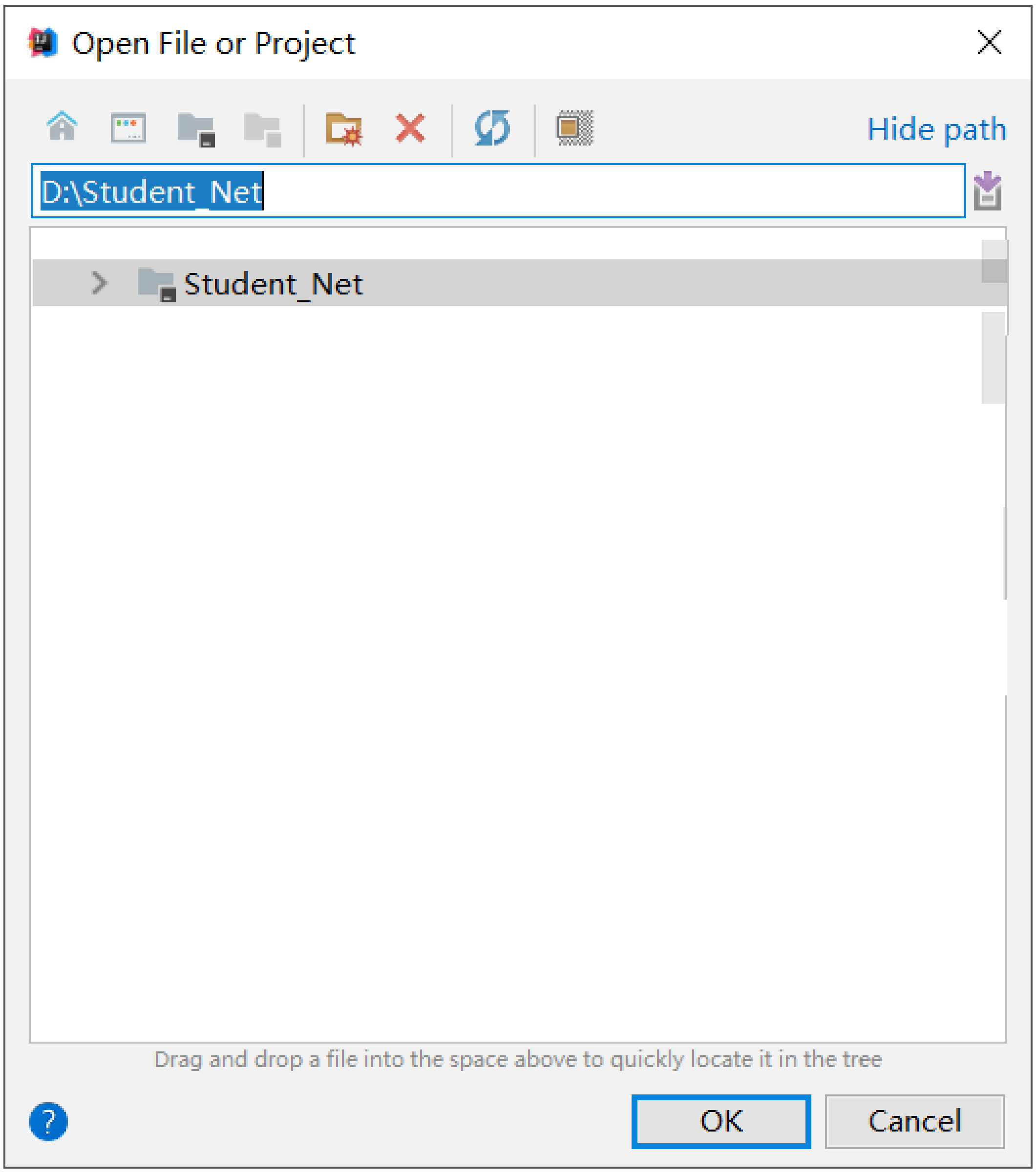
一 项目演示

1 打开项目

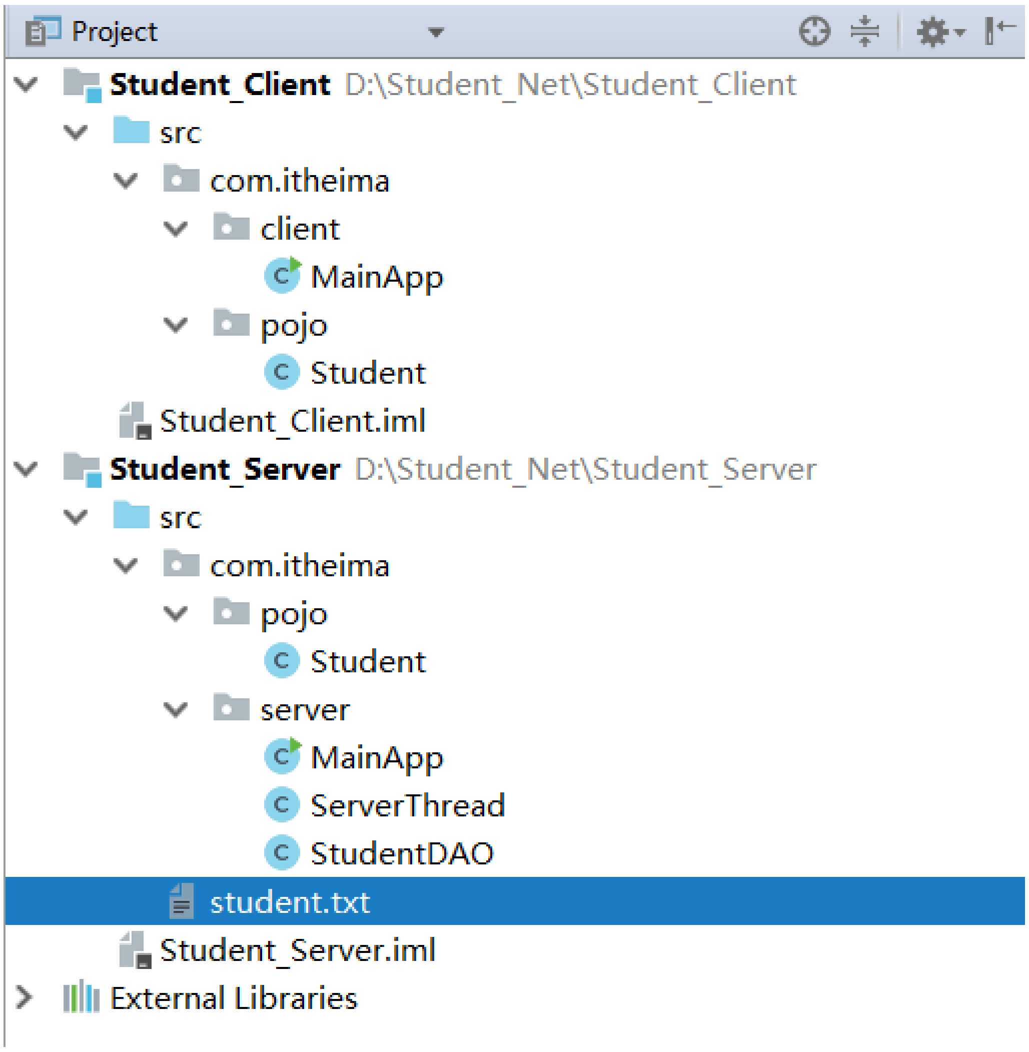
- 将演示程序复制到本地磁盘


Student_Net

- 启动IDEA，选择：File --> open，在"打开对话框"中选择这个项目目录


Open File or Project
Hide path
D:\Student_Net
Student_Net
Drag and drop a file into the space above to quickly locate it in the tree
OK
Cancel

- 目录结构如下图：

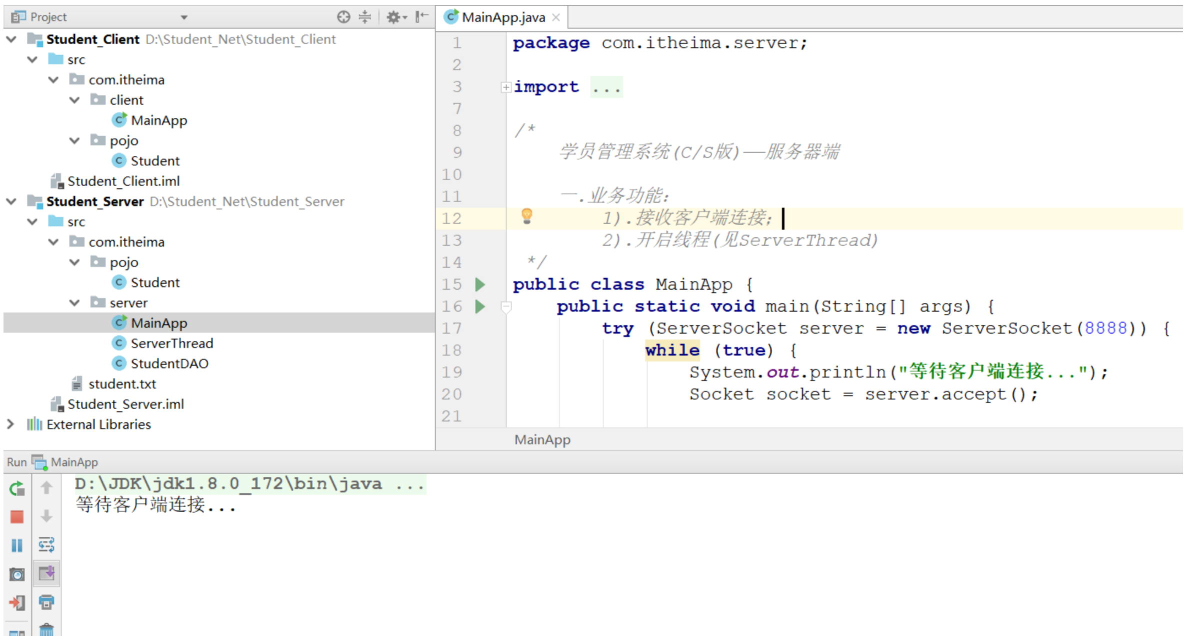


```
Project
Student_Client D:\Student_Net\Student_Client
  src
    com.itheima
      client
        MainApp
      pojo
        Student
  Student_Client.iml
Student_Server D:\Student_Net\Student_Server
  src
    com.itheima
      pojo
        Student
      server
        MainApp
        ServerThread
        StudentDAO
    student.txt
  Student_Server.iml
External Libraries
```

2 运行项目

- 运行服务器端：Student_Server/com/itheima/server/MainApp：

注：服务器使用端口：8888



```
Project
Student_Client D:\Student_Net\Student_Client
  src
    com.itheima
      client
        MainApp
      pojo
        Student
  Student_Client.iml
Student_Server D:\Student_Net\Student_Server
  src
    com.itheima
      pojo
        Student
      server
        MainApp
        ServerThread
        StudentDAO
    student.txt
  Student_Server.iml
External Libraries
```

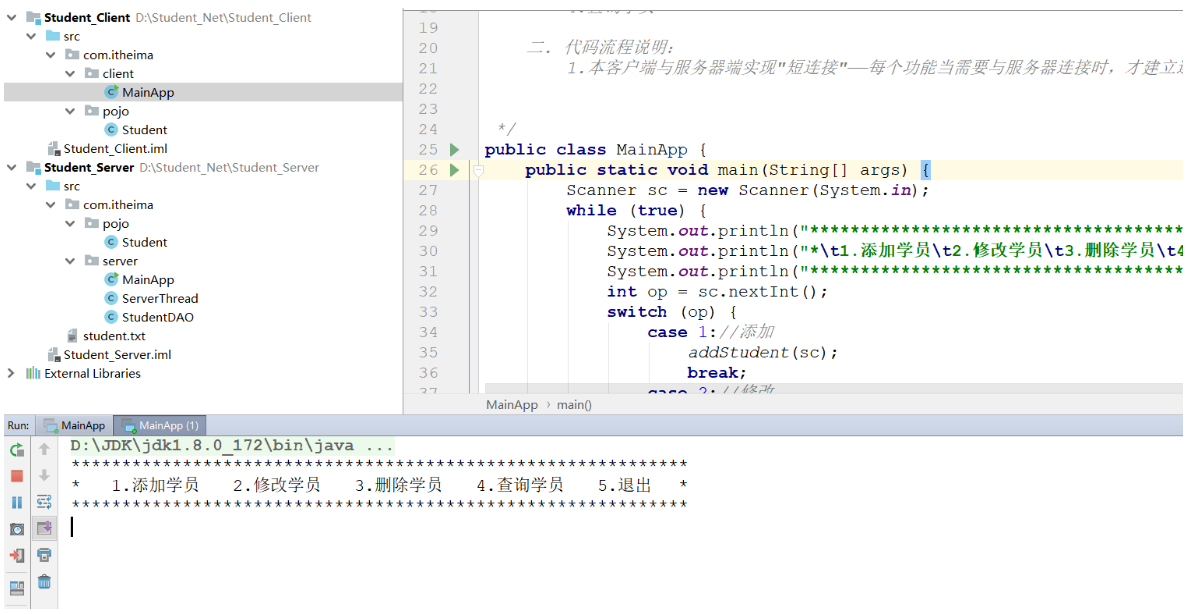
MainApp.java

```java
package com.itheima.server;

import ...

/*
    学员管理系统(C/S版)——服务器端

    一.业务功能:
        1).接收客户端连接;
        2).开启线程(见ServerThread)
 */
public class MainApp {
    public static void main(String[] args) {
        try (ServerSocket server = new ServerSocket(8888)) {
            while (true) {
                System.out.println("等待客户端连接...");
                Socket socket = server.accept();
```

Run: MainApp

```
D:\JDK\jdk1.8.0_172\bin\java ...
等待客户端连接...
```

- 运行客户端：Student_Client/com/itheima/client/MainApp：



```

     二. 代码流程说明:
         1.本客户端与服务器端实现"短连接"——每个功能当需要与服务器连接时，才建立


 */
public class MainApp {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        while (true) {
            System.out.println("************************************
            System.out.println("*\t1.添加学员\t2.修改学员\t3.删除学员\t4
            System.out.println("************************************
            int op = sc.nextInt();
            switch (op) {
                case 1://添加
                    addStudent(sc);
                    break;
```

```
D:\JDK\jdk1.8.0_172\bin\java ...
****************************************************************
*    1.添加学员    2.修改学员    3.删除学员    4.查询学员    5.退出    *
****************************************************************
|
```

- 执行添加、修改、删除、查询：

注：文件在%项目目录%\out\production\Student_Server\student.txt

二 项目说明

1 所采用的知识点

本系统采用了我们学过的以下几个核心知识点：

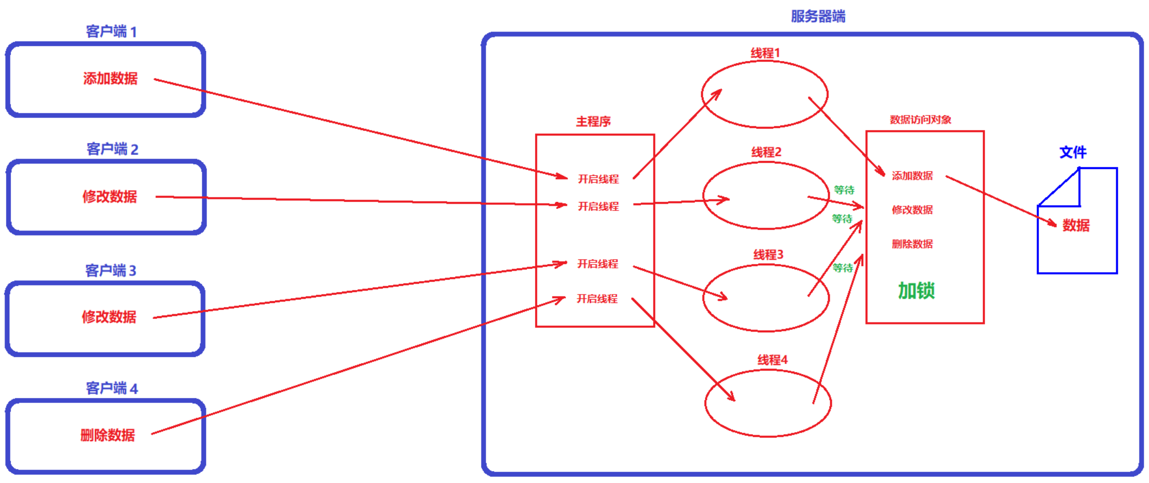
1). IO流技术

2). 网络编程技术

3). 序列化

4). 多线程

2 业务交互模式图示



【说明】

1).客户端和服务器端采用TCP连接；

2).数据保存在服务器端；

3). 客户端增删改查发送数据格式说明：

a). 添加："[1]数据"，例如："[1]张三,男,22"，意思：没有id字段，由服务器端在写入数据前自动添加。

b).根据id查询一条数据："[2]id"，例如："[2]1"，意思：查询id为1的学员信息

c). 修改一条数据："[3]新数据"。例如："[3]1,张三2,女,19"，意思：将id=1的学员改为后面的新数据。

d). 查询所有数据："[4]"。例如："[4]"，意思：后面不用带任何数据。

e). 删除一条数据："[5]id"。例如："[5]1"，意思：删除id为1的记录。

三 案例代码

1 客户端

- 创建实体类：com.itheima.pojo.Student类：

```java
package com.itheima.pojo;

import java.io.Serializable;

public class Student implements Serializable {
    private int id;
    private String name;
    private String sex;
    private int age;

    public Student() {
    }

    public Student(int id, String name, String sex, int age) {
        this.id = id;
        this.name = name;
        this.sex = sex;
        this.age = age;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }
    public String getSex() {
        return sex;
    }

    public void setSex(String sex) {
        this.sex = sex;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    @Override
    public String toString() {
        return "Student{" +
                "id=" + id +
                ", name='" + name + '\'' +
                ", sex='" + sex + '\'' +
                ", age=" + age +
                '}';
    }
}
```

- 创建主类：com.itheima.client.MainApp类：

```java
package com.itheima.client;

import com.itheima.pojo.Student;

import java.io.*;
import java.net.Socket;
import java.net.UnknownHostException;
import java.util.Scanner;
import java.util.ArrayList;

/*
    学员管理系统(C/S版)--客户端

    一. 业务功能：
        1.添加学员
        2.修改学员
        3.删除学员
        4.查询学员

    二. 代码流程说明：
        1.本客户端与服务器端实现"短连接"--每个功能当需要与服务器连接时，才建立连接，功能完毕，
连接立即断开；
```

```
 */
public class MainApp {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        while (true) {

 System.out.println("*****************************************************");
            System.out.println("*\t1.添加学员\t2.修改学员\t3.删除学员\t4.查询学员\t5.
退出\t*");

 System.out.println("*****************************************************");
            int op = sc.nextInt();
            switch (op) {
                case 1://添加
                    addStudent(sc);
                    break;
                case 2://修改
                    updateStudent(sc);
                    break;
                case 3://删除
                    deleteStudent(sc);
                    break;
                case 4://查询
                    findStudent(sc);
                    break;
                case 5://退出
                    System.out.println("谢谢使用，再见！");
                    System.exit(0);
            }

        }
    }

    //1.添加学员
    private static void addStudent(Scanner sc) {
        //1.接收用户数据
        System.out.println("请输入学员信息：");
        System.out.println("姓名：");
        String name = sc.next();
        System.out.println("性别：");
        String sex = sc.next();
        System.out.println("年龄：");
        int age = sc.nextInt();


        //2.获取连接后的输出流
        Socket socket = getSocket();
        if (socket == null) {
            System.out.println("【错误】无法连接服务器！");
            return;
        }
        //3.创建输出流
        try(OutputStream netOut = socket.getOutputStream();
            InputStream netIn = socket.getInputStream();
        ) {
            //发送数据
            netOut.write(("[1]" +  name + "," + sex + "," + age).getBytes());
            //接收反馈
            int b = netIn.read();
            //4.关闭连接
            socket.close();
            //判断反馈
            if(b == 0){
                //5.完毕
                System.out.println("【成功】数据已保存！");
            }else{
                System.out.println("【失败】数据保存失败，请重试！");
            }
            return;
        } catch (IOException e) {
            System.out.println("【错误】保存失败，请重试！");
            return;
        }
    }



    //2.修改学员
    private static void updateStudent(Scanner sc) {
        //1.接收id
        System.out.println("请输入要修改的学员ID：");
        int id = sc.nextInt();

        //2.获取连接
        Socket socket = getSocket();
        //3.发送"查询"请求
        try {
            OutputStream netOut = socket.getOutputStream();
            InputStream netIn = socket.getInputStream();
            //标记："2"根据ID查询一条记录
            netOut.write(("[2]" + id).getBytes());
            //接收结果
            ObjectInputStream objIn = new ObjectInputStream(netIn);
            Object obj = objIn.readObject();
            objIn.close();

            if (obj == null) {
                System.out.println("【失败】无查询结果！");
                return;
            }
            if (!(obj instanceof Student)) {
                System.out.println("【失败】返回数据错误，请重试！");
                return;
            }
            //关闭此次连接
            socket.close();

            //向下转型
            Student stu = (Student)obj;
            System.out.println("【查询结果】");
            printStudent(stu);//打印

            //接收新数据
            System.out.println("请输入新姓名(保留原值请输入0)：");
            String newName = sc.next();
            System.out.println("请输入新性别(保留原值请输入0)：");
            String newSex = sc.next();
            System.out.println("请输入新年龄(保留原值请输入0)：");
            int newAge = sc.nextInt();

            if (!"0".equals(newName)) {
                stu.setName(newName);
            }
            if (!"0".equals(newSex)) {
                stu.setSex(newSex);
            }
            if (newAge != 0) {
                stu.setAge(newAge);
            }
            //再次连接
            socket = getSocket();
            //发送修改数据，格式：[3]....
            netOut = socket.getOutputStream();
            netOut.write(("[3]" + stu.getId() + "," +
                          stu.getName() + "," +
                          stu.getSex() + "," +
                          stu.getAge()).getBytes());
            //接收反馈
            netIn = socket.getInputStream();
            int b = netIn.read();
            if (b == 0) {
                System.out.println("【成功】数据已修改！");
            }else{
                System.out.println("【失败】数据修改失败，请重试！");
            }
            return;

        } catch (IOException e) {
            e.printStackTrace();
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }

    }
    //3.删除学员
    private static void deleteStudent(Scanner sc) {
        System.out.println("请输入要删除的学员ID：");
        int id = sc.nextInt();
        //2.获取连接
        Socket socket = getSocket();
        //3.发送"查询"请求
        try {
            OutputStream netOut = socket.getOutputStream();
            InputStream netIn = socket.getInputStream();
            //标记："2"根据ID查询一条记录
            netOut.write(("[2]" + id).getBytes());


            ObjectInputStream objIn = new ObjectInputStream(netIn);
            //接收结果
            Object obj = objIn.readObject();
            if (obj == null) {
                System.out.println("【失败】无查询结果！");
                return;
            }
            if (!(obj instanceof Student)) {
                System.out.println("【失败】返回数据错误，请重试！");
                return;
            }
            //向下转型
            Student stu = (Student)obj;
            System.out.println("【查询结果】");
            printStudent(stu);//打印

            //关闭连接
            socket.close();

            //确认删除
            System.out.println("【确认】你确定删除这条记录吗？(y/n)：");
            String op = sc.next();
            if (!"y".equals(op)) {
                System.out.println("【取消】操作被取消！");
                return;
            }

            //再次连接
            socket = getSocket();
            //发送删除数据，格式：[5]id值....
            netOut = socket.getOutputStream();
            netOut.write(("[5]" + stu.getId()).getBytes());
            //接收反馈
            netIn = socket.getInputStream();
            int b = netIn.read();
            if (b == 0) {
                System.out.println("【成功】数据已删除！");
            }else{
                System.out.println("【失败】数据删除失败，请重试！");
            }
            return;

        } catch (IOException e) {
            e.printStackTrace();
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }

    }
    //4.查询学员
    private static void findStudent(Scanner sc) {
        //
        //1.获取连接
        Socket socket = getSocket();

        try{
            OutputStream netOut = socket.getOutputStream();
            //2.发送请求，格式：[4]
            netOut.write(("[4]").getBytes());

            ObjectInputStream objIn = new ObjectInputStream(
                                                socket.getInputStream());
            //3.接收结果，一个序列化的ArrayList<Student>
            Object o = objIn.readObject();
            if (o == null) {
                System.out.println("【失败】查询失败，请重试！");
                return;
            }
            if (!(o instanceof ArrayList)) {
                System.out.println("【错误】返回数据错误，请重试！");
                return;
            }

            System.out.println("【查询结果】");
            ArrayList<Student> list = (ArrayList<Student>)o;
            printStudentList(list);
            //关闭连接
            socket.close();


        } catch (IOException e) {
            e.printStackTrace();
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
    }

    //连接服务器
    private static Socket getSocket(){
        String ip = "127.0.0.1";
        int port = 8888;

        try {
            Socket socket = new Socket(ip, port);
            return socket;
        } catch (UnknownHostException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return null;
    }



    //打印ArrayList<Student>的方法
    public static void printStudentList(ArrayList<Student> stuList) {
        System.out.println("--------------------------------------------------
");
        System.out.println("编号\t\t姓名\t\t\t性别\t\t年龄");
        for (int i = 0; i < stuList.size(); i++) {
            Student p = stuList.get(i);
            System.out.println(p.getId() + "\t\t" +
                               p.getName() + "\t\t\t" +
                               p.getSex() + "\t\t" +
                               p.getAge());

        }
        System.out.println("--------------------------------------------------
");
    }
    //打印Person的方法
    public static void printStudent(Student stu) {
        System.out.println("--------------------------------------------------
");
        System.out.println("编号\t\t姓名\t\t性别\t\t\t年龄");
        System.out.println(stu.getId() + "\t\t" +
                           stu.getName() + "\t\t\t" +
                          stu.getSex() + "\t\t" +
                          stu.getAge());
        System.out.println("--------------------------------------------------
");
    }




}
```

2 服务器端

- 创建实体类：com.itheima.pojo.Student类：

```
package com.itheima.pojo;

import java.io.Serializable;

public class Student implements Serializable {
    private int id;
    private String name;
    private String sex;
    private int age;

    public Student() {
    }

    public Student(int id, String name, String sex, int age) {
        this.id = id;
        this.name = name;
        this.sex = sex;
        this.age = age;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getSex() {
        return sex;
    }

    public void setSex(String sex) {
        this.sex = sex;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    @Override
    public String toString() {
        return "Student{" +
                "id=" + id +
                ", name='" + name + '\'' +
                ", sex='" + sex + '\'' +
                ", age=" + age +
                '}';
    }
}
```

- 创建服务器端线程类：com.itheima.server.ServerThread类：

```
package com.itheima.server;

import com.itheima.pojo.Student;

import java.io.IOException;
import java.io.InputStream;
import java.io.ObjectOutputStream;
import java.io.OutputStream;
import java.net.Socket;
import java.util.ArrayList;

/*
    服务器端线程：

    一.业务功能：
        1).接收客户端增、删、改、查的请求；
        2).调用StudentDAO处理增、删、改、查的业务；
        3).为客户端返回处理结果

 */
public class ServerThread extends Thread {
```

```java
    private Socket socket;//与客户端连接的Socket对象
    public ServerThread(Socket socket) {
        this.socket = socket;
    }

    @Override
    public void run() {
        try (InputStream netIn = this.socket.getInputStream();
             OutputStream netOut = this.socket.getOutputStream();
        ) {
            //接收客户端数据
            byte[] bytes = new byte[1024];
            int len = netIn.read(bytes);//只接收一次，最多1K
            String msg = new String(bytes, 0, len);
            if (msg.charAt(0) != '[' ||
                    msg.indexOf("]") == -1) {
                //关闭连接
                System.out.println("未知数据格式，线程结束！");
                socket.close();
                return;
            }
            //解析标记位
            String flag = msg.substring(0 + 1,msg.indexOf("]"));
            //判断
            switch (flag) {
                case "1"://添加
                    addStudent(msg);
                    break;
                case "2"://根据id查询一条
                    System.out.println("查询一条");
                    findById(msg);
                    break;
                case "3"://修改一条

                    updateStudent(msg);
                    break;
                case "4"://查询所有
                    findAll(msg);
                    break;
                case "5"://删除一条
                    deleteById(msg);
                    break;
                default:
                    System.out.println("未知数据格式！");
                    socket.close();
                    break;

            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    //删除一条
    private void deleteById(String msg) {
        msg = msg.substring(msg.indexOf("]") + 1);

        int id = Integer.parseInt(msg);
        boolean b = StudentDAO.deleteById(id);

        try{
            OutputStream netOut = socket.getOutputStream();
            if (b) {
                netOut.write(0);
            }else{
                netOut.write(1);
            }
            socket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    private void findAll(String msg) {
        ArrayList<Student> all = StudentDAO.findAll();

        //直接序列化集合给客户端
        try  {
            ObjectOutputStream objOut = new ObjectOutputStream(
                                                socket.getOutputStream());
            objOut.writeObject(all);

            //关闭连接
            socket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    //处理根据ID查询
    private void findById(String msg) {
        msg = msg.substring(msg.indexOf("]") + 1);
        int id = Integer.parseInt(msg);
        Student stu = StudentDAO.findById(id);
        try{
            OutputStream netOut = socket.getOutputStream();
            //直接序列化给客户端
            ObjectOutputStream objOut = new ObjectOutputStream(
                                                socket.getOutputStream());
            System.out.println("序列化");
            objOut.writeObject(stu);
            System.out.println("序列化完毕");


            //关闭连接
            socket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }

    }


    //处理修改
    private void updateStudent(String msg) {
        msg = msg.substring(msg.indexOf("]") + 1);//"1,张三,男,22"
        String[] arr = msg.split(",");
        Student stu = new Student();
        stu.setId(Integer.parseInt(arr[0]));
        stu.setName(arr[1]);
        stu.setSex(arr[2]);
        stu.setAge(Integer.parseInt(arr[3]));

        boolean b = StudentDAO.updateStudent(stu);
        try (OutputStream netOut = socket.getOutputStream()) {
            if (b) {
                netOut.write(0);
            }else{
                netOut.write(1);
            }
            socket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }

    }
    //处理添加
    private void addStudent(String msg) {
        msg = msg.substring(msg.indexOf("]") + 1);//"张三,男,22"
        String[] arr = msg.split(",");
        Student stu = new Student();
        stu.setName(arr[0]);
        stu.setSex(arr[1]);
        stu.setAge(Integer.parseInt(arr[2]));

        boolean b = StudentDAO.addStudent(stu);

        //返回给客户端处理结果
        try  {
            OutputStream netOut = socket.getOutputStream();
            if (b) {
                netOut.write(0);
            }else{
                netOut.write(1);
            }
            socket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

- 创建服务器端主类：com.itheima.server.MainApp类：

```
package com.itheima.server;

import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;
import java.nio.channels.ServerSocketChannel;
```

```java
/*
    学员管理系统(C/S版)--服务器端

    一.业务功能:
        1).接收客户端连接;
        2).开启线程(见ServerThread类)
 */
public class MainApp {
    public static void main(String[] args) {
        try (ServerSocket server = new ServerSocket(8888)) {
            while (true) {
                System.out.println("等待客户端连接...");
                Socket socket = server.accept();
                //开启线程
                new ServerThread(socket).start();
            }

        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```