1、简介

本次作业调用前一组（17 组）的验证服务以及后一组（18 组）的分数和学生管理服务，完成客户端的调用。
前一组服务 wsdl：http://115.159.202.18:3323/ServiceHello?wsdl
后一组服务 wsdl：http://106.15.91.25:8080/grade/cxf/scoreManageService?wsdl
http://106.15.91.25:8080/grade/cxf/studentInfoService?wsdl

2、作业步骤与要点

1）使用 wsdl2java 生成 Java 接口文件

使用 cxf 的 wsdl2java 工具，输入命令：
wsdl2java –encoding utf-8 –d {directory} {wsdl url}

2）编写 handler

公有 4 个 handler，分别是验证密码的 AuthHandler，验证老师身份的 AuthIdentifyHandler，保存所有输入输出消息的 LogHandler，以及修改分数时同步修改学生信息的 StudentModifyHandler。

a． 身份验证

以身份验证 AuthIdentifyHandler 为例，获取发送请求时设置在 SoapHeader 中的 email 和密码，调用前一组的身份验证服务验证身份。

```
if ((Boolean) context.get(SOAPMessageContext.MESSAGE_OUTBOUND_PROPERTY)) {
    SOAPMessage soapMessage = context.getMessage();
    try {
        SOAPHeader soapHeader = soapMessage.getSOAPHeader();
        if (soapHeader!=null) {
            NodeList nodeList = soapHeader.getChildNodes();
            String email = nodeList.item( index: 0).getTextContent();
            String password = nodeList.item( index: 1).getTextContent();
            MyService myService = new MyService();
            myService.setHandlerResolver(new DefaultResolver());
            HelloServiceInterface helloServiceInterface = myService.getHelloServicePort();
            Login login = new Login();
            login.setLoginUsername(email);
            login.setLoginPassword(password);
            try {
                LoginResponse response = helloServiceInterface.login(login);
                if (!response.getLoginResults().getKind().equals("教师")) {
                    writeMsg("无权限操作.");
                    return false;
                }
            } catch (LoginException e) {
                writeMsg(e.getFaultInfo().getMessage());
                return false;
            }
        }
    } catch (SOAPException e) {
```

b． 保存输入输出

不管是正常消息还是出错消息，都保存到文件

```
@Override
public boolean handleMessage(SOAPMessageContext context) {
    recordMsg(context);
    return true;
}

@Override
public boolean handleFault(SOAPMessageContext context) {
    recordMsg(context);
    return true;
}
```

文件读写片段：

```
SOAPMessage soapMessage = context.getMessage();
try {
    File file = new File( pathname: "msg.log");
    if (!file.exists()) {
        boolean result = file.createNewFile();
        if (!result)
            throw new IOException("Cannot create file");
    }
    FileOutputStream outputStream = new FileOutputStream(file, append: true);
    if (!(Boolean) context.get(SOAPMessageContext.MESSAGE_OUTBOUND_PROPERTY)) {
        outputStream.write("====================RESPONSE====================\n".getBytes());
    } else {
        outputStream.write("====================REQUEST====================\n".getBytes());
    }
    TransformerFactory tff = TransformerFactory.newInstance();
    Transformer tf = tff.newTransformer();
    // Set formatting

    tf.setOutputProperty(OutputKeys.INDENT,  value: "yes");
    tf.setOutputProperty( name: "{http://xml.apache.org/xslt}indent-amount",
            value: "2");

    Source sc = soapMessage.getSOAPPart().getContent();

    ByteArrayOutputStream streamOut = new ByteArrayOutputStream();
    StreamResult result = new StreamResult(streamOut);
    tf.transform(sc, result);

    String strMessage = streamOut.toString();
    outputStream.write(strMessage.getBytes());
```

c． 同步对学生的修改

这个 handler 从修改分数成功后的 soap 消息中读取修改的分数信息，根据学号获取学生信息，修改对应的学生成绩的信息。
从返回消息中获取修改成功的分数信息，并调用服务获取学生信息：

```
if (!(Boolean) context.get(SOAPMessageContext.MESSAGE_OUTBOUND_PROPERTY)) {
    StudentInfoServiceImplService serviceImplService = new StudentInfoServiceImplService();
    serviceImplService.setHandlerResolver(new DefaultResolver());
    StudentInfoService service = serviceImplService.getStudentInfoServicePort();
    try {
        SOAPBody soapBody = context.getMessage().getSOAPBody();
        Node courseScoreList = soapBody.getFirstChild();
        NodeList courseScores = courseScoreList.getChildNodes();
        for (int i = 0; i < courseScores.getLength(); i++) {
            Node courseScore = courseScores.item(i);
            String cid = courseScore.getAttributes().getNamedItem("课程编号").getTextContent();
            String ctype = courseScore.getAttributes().getNamedItem("成绩性质").getTextContent();
            NodeList scores = courseScore.getChildNodes();
            for (int j = 0; j < scores.getLength(); j++) {
                Node score = scores.item(j);
                String sid = score.getChildNodes().item( index: 0).getTextContent();
                String scoreStr = score.getChildNodes().item( index: 1).getTextContent();
                课程成绩类型 courseScoreType = new 课程成绩类型();
                courseScoreType.set课程编号(cid);
                courseScoreType.set成绩性质(成绩性质类型.fromValue(ctype));
                List<成绩类型> scoreTypes = new ArrayList<>();
                成绩类型 scoreType = new 成绩类型();
                scoreType.set学号(sid);
                scoreType.set得分(Integer.parseInt(scoreStr));
                scoreTypes.add(scoreType);
                courseScoreType.set成绩(scoreTypes);
                学生信息 studentInfo = packageStudentInfo(service.queryInfo(sid), courseScoreType);
                service.modifyInfo(new Holder<>(studentInfo));
            }
```

获取学生信息

修改更新分数后的学生信息

3） 编写 resolver

编写调用不同操作时组合了不同 handler 的 resolver，DefaultResolver 只有日志记录，SimpleResolver 有日志和简单的身份验证，ComplexResolver 有日志、身份验证和学生信息同步。

```
public class ComplexResolver implements HandlerResolver{
    @Override
    public List<Handler> getHandlerChain(PortInfo portInfo) {
        List<Handler> handlers = new ArrayList<>();
        handlers.add(new LogHandler());
        handlers.add(new AuthIdentityHandler());
        handlers.add(new StudentModifyHandler());
        return handlers;
    }
}
```

4） 调用服务

注意在调用服务时设置 header

```
public void addScore(课程成绩列表类型 listType, String email, String password) {
    ScoreManageServiceImplService service = new ScoreManageServiceImplService();
    service.setHandlerResolver(new ComplexResolver());
    ScoreManageService scoreManageService = service.getScoreManageServicePort();
    WSBindingProvider bp = (WSBindingProvider)scoreManageService;
    bp.setOutboundHeaders(Headers.create(new QName(WsdlUrl.AUTH_NS, localPart: "email"),email),
            Headers.create(new QName(WsdlUrl.AUTH_NS, localPart: "password"),password));
    Holder<课程成绩列表类型> param = new Holder<>(listType);
    try {
        scoreManageService.addGrade(param);
    } catch (InvalidCourseId | InvalidStudentId | InvalidScore invalidCourseId) {
        invalidCourseId.printStackTrace();
    }
}
```

5）覆盖用例

覆盖身份验证失败、不是老师身份、学生 ID 不合法等输入参数问题、以及正常调用各个服务的用例。

```
//删除成绩
学号课程号类型 sidCid = new 学号课程号类型();
sidCid.set学号("141250179");
sidCid.set课程编号("10001");
service.deleteScore(sidCid, email: "141250019@smail.nju.edu.cn", password: "0194");
service.deleteScore(sidCid, email: "teacher@nju.edu.cn", password: "123456");

//查询成绩
service.queryScore( sid: "141250179", email: "141250019@smail.nju.edu.cn", password: "123456");
service.queryScore( sid: "141250029", email: "teacher@nju.edu.cn", password: "123456");
service.queryScore( sid: "141250179", email: "teacher@nju.edu.cn", password: "123456");
```

这里的方法是已经封装好的服务调用方法，服务调用的细节方法被封装在这个方法里。

6）对错误的处理

在分数修改同步学生信息发生错误时不能进行完整的调用流程，所有的错误消息和错误提示都会被保存到 msg.log 中。

```
====================RESPONSE====================
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <S:Fault xmlns:ns4="http://www.w3.org/2003/05/soap-envelope">
      <faultcode>S:Server</faultcode>
      <faultstring>账号不存在或者密码错误</faultstring>
      <detail>
        <ns2:LoginException xmlns:ns2="http://service/" xmlns:ns3="http://mail/login">
          <message>账号不存在或者密码错误</message>
        </ns2:LoginException>
      </detail>
    </S:Fault>
  </S:Body>
</S:Envelope>

====================Handler Info====================
账号不存在或者密码错误
```

```
} catch (LoginException e) {
    try {
        File file = new File( pathname: "msg.log");
        if (!file.exists()) {
            boolean result = file.createNewFile();
            if (!result)
                throw new IOException("Cannot create file");
        }
        FileOutputStream outputStream = new FileOutputStream(file, append: true);
        outputStream.write("====================Handler Info====================\n".getBytes());
        outputStream.write(e.getFaultInfo().getMessage().getBytes());
        outputStream.write( b: '\n');
    } catch (IOException e1) {
        e1.printStackTrace();
    }
    return false;
}
```

服务定义的返回异常

7） Namespace 差异

前一组和后一组 namespace 的差异处理，体现在生成 java 文件时不同的类型在不同的 java package 下，并用 package-info.java 指定类型的命名空间，在编写 java 代码时，通过指定不同的 package 就可以处理 namespace 差异。

```
@javax.xml.bind.annotation.XmlSchema(namespace = "http://jw.nju.edu.cn/wsdl")
package cn.edu.nju.jw.wsdl;
```

在生成类型时指定命令空间：

```
@WebFault(name = "学号错误", targetNamespace = "http://jw.nju.edu.cn/schema")
public class InvalidStudentId extends Exception {
```

```
 */
@WebService(targetNamespace = "http://jw.nju.edu.cn/wsdl", name = "StudentInfoService")
@XmlSeeAlso({cn.edu.nju.schema.ObjectFactory.class, cn.edu.nju.jw.schema.ObjectFactory.class, ObjectFactory.class})
@SOAPBinding(parameterStyle = SOAPBinding.ParameterStyle.BARE)
public interface StudentInfoService {
```