



SOA 作业报告

作业 10



2017-6-15

南京大学软件学院
18 组

目录

1、 作业简介	2
2、具体步骤	2
(1) 转换作业 7、8 的 wsdl	2
(2) 从 wsdl 生成 java 文件	2
(3) 完善 model 与 entity	3
a、refactor	3
b、数据库建表	3
c、生成映射 entity	3
(4) 数据库 JPA	4
(5) 发布 webservice	5
(6) 配置 webservice	5
(7) 运行项目	6
(8) 测试接口	6
(8) 发布服务	9

1、作业简介

本次作业使用 SpringBoot + Apache CXF 为 webservice 框架，发布于 springboot 自带的 tomcat 服务器上，使用了 Mysql 数据库，并使用了 Spring Data JPA 进行 ORM，使用 maven 构建项目。

虽然 SOAP 是一种比较古老的技术，但仍然可以使用最新的平台和技术发挥它的作用。服务已经发布在服务器：<http://106.15.183.234:8080/soap-api>

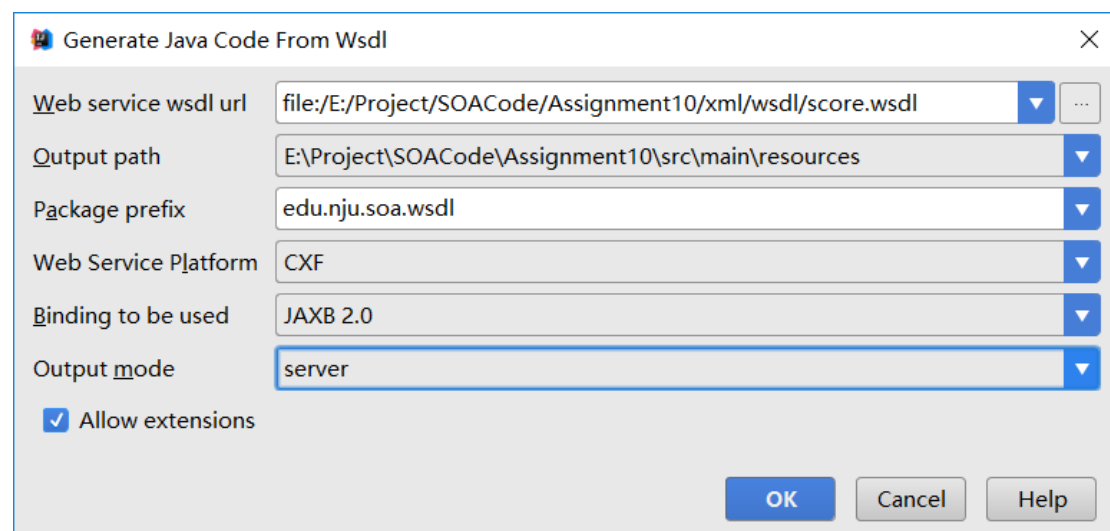
2、具体步骤

(1) 转换作业 7、8 的 wsdl

因为作业 7、8 使用的 wsdl 是 2.0 的，而 start from wsdl 转换工具 wsdl2java 不支持 2.0 的转换，因此首先需要将之前作业的 wsdl 文件转换成 1.2 的版本。

转换的文件位于/xml/wsdl 文件夹，为了转换需要新增了一些输入类型。

(2) 从 wsdl 生成 java 文件



使用本地文件生成 wsdl 文件

(3) 完善 model 与 entity

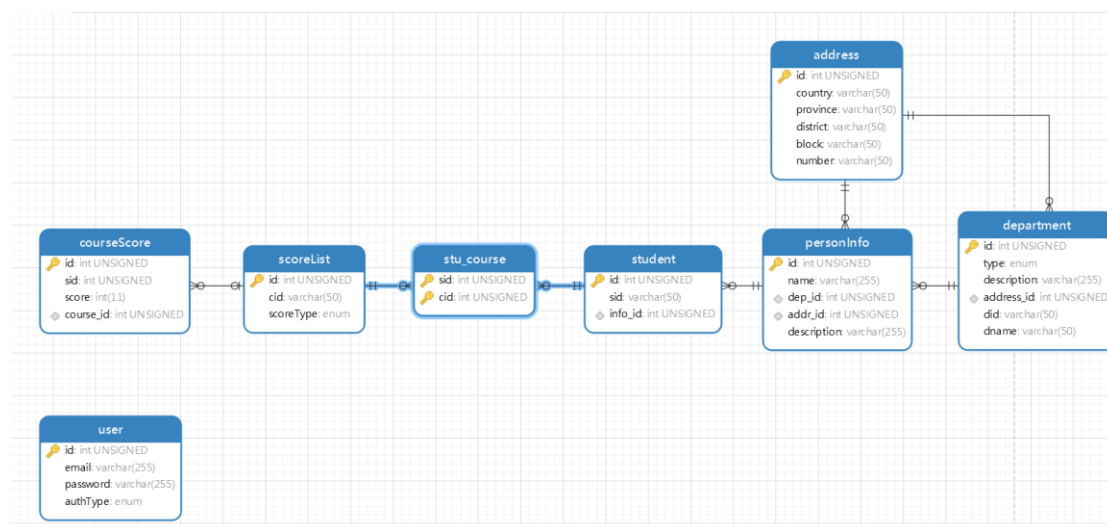
a、refactor

因为之前定义的 schema 为中文，但是因为编码不方便，所以将类和属性的名字改成中文，但是保持定义 schema 的中文名字不变。

```
33 *
34 *
35 */
36 @XmlAccessorType(XmlAccessType.FIELD)
37 @XmlType(name = "课程成绩类型", propOrder = {
38     "courseScoreTypes"
39 })
40 @NoArgsConstructor
41 @AllArgsConstructor
42 public class CourseScore {
43
44     @XmlElement(name = "成绩")
45     protected List<CourseScoreType> courseScoreTypes;
46     @XmlAttribute(name = "课程编号")
47     protected String cid;
48     @XmlAttribute(name = "成绩性质")
49     protected ScoreType scoreType;
```

b、数据库建表

为了映射方便，应当保持表结构与 schema 定义类似。



c、生成映射 entity

自动生成一部分代码，手动添加表关联（One to many, Many to Many 等）注解。

- ③ AddressEntity
- ③ CourseScoreEntity
- ③ DepartmentEntity
- ③ PersonInfoEntity
- ③ ScoreListEntity
- ③ StudentEntity
- ③ UserEntity

```

@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
@Column(name = "id", nullable = false)
public int getId() { return id; }

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

@ManyToOne(fetch = FetchType.EAGER)
@JoinColumn(name = "info_id", referencedColumnName = "id")
public PersonInfoEntity getPersonInfo() { return personInfo; }

public void setPersonInfo(PersonInfoEntity personInfo) { this.personInfo = personInfo; }

@ManyToMany
@JoinTable(name = "stu_course", schema = "soa",
    joinColumns = @JoinColumn(name = "sid", referencedColumnName = "id", nullable = false),
    inverseJoinColumns = @JoinColumn(name = "cid", referencedColumnName = "id", nullable = false))
public List<ScoreListEntity> getScoreList() { return scoreList; }

```

(4) 数据库 JPA

定义 Spring Data JPA 接口，比如：

```

/**
 * Created by cuihao on 2017-06-13.
 * Auth repo definition
 */
public interface UserRepository extends JpaRepository<UserEntity,Integer> {

    UserEntity findByIdEmail(String email);

}

```

定义并实现 DAO

```

@Repository
public class UserDaoImpl implements UserDao {

    @Resource
    private UserRepository userRepository;

    @Override
    public UserEntity findById(int id) { return userRepository.findOne(id); }

    @Override
    public UserEntity findByIdEmail(String email) { return userRepository.findByIdEmail(email); }

    @Override
    public void delete(int id) { userRepository.delete(id); }

    @Override
    public UserEntity save(UserEntity userEntity) { return userRepository.save(userEntity); }

}

```

```

@Override
@Transactional
public StudentEntity save(StudentEntity studentEntity) {
    PersonInfoEntity personInfo = studentEntity.getPersonInfo();
    DepartmentEntity departmentEntity = personInfo.getDepartment();
    departmentEntity.setAddress(addressRepository.save(departmentEntity.getAddress()));
    personInfo.setDepartment(departmentRepository.save(departmentEntity));
    personInfo.setAddress(addressRepository.save(personInfo.getAddress()));
    studentEntity.setPersonInfo(personInfoRepository.save(personInfo));
    List<ScoreListEntity> scoreListEntities = studentEntity.getScoreList();
    if (scoreListEntities != null && !scoreListEntities.isEmpty()) {
        studentEntity.setScoreList(scoreListEntities.stream().map(s -> scoreDao.save(s)).collect(Collectors.toList()));
    }
    return studentRepository.save(studentEntity);
}

```

(5) 发布 webservice

作业 7、8 的接口使用生成的 java 文件中的接口，邮箱验证接口自己定义，然后实现这个 web service 接口。

需要注意的是，同一门课程并且成绩类型相同时应该将很多成绩放在一个课程的标签里。

示例：

```

/**
 * Add score
 * @param courseScoreListTypeHolder Mode: in and out
 * @throws IdNotFoundException cannot find id
 * @throws ScoreTypeException score type error
 * @throws ScoreModifyException score modify error
 * @throws AuthorityException do not have authority
 */
@WebMethod(action = "score/addScore")
@RequestWrapper(localName = "addScore", targetNamespace = "http://jw.nju.edu.cn/schema", className = "cn.edu.nju.jw.
@ResponseWrapper(localName = "modifyScore", targetNamespace = "http://jw.nju.edu.cn/schema", className = "cn.edu.nju
@Action(input = "score/addScoreRequest", output = "score/addScoreResponse", fault = {
    @FaultAction(className = IdNotFoundException.class, value = "score/addScore/Fault/IdNotFoundException"),
    @FaultAction(className = ScoreTypeException.class, value = "score/addScore/Fault/ScoreTypeException"),
    @FaultAction(className = ScoreModifyException.class, value = "score/addScore/Fault/ScoreModifyException"),
    @FaultAction(className = AuthorityException.class, value = "score/addScore/Fault/AuthorityException")
})
void addScore(
    @WebParam(name = "课程成绩列表", targetNamespace = "http://jw.nju.edu.cn/schema", mode = WebParam.Mode.INOUT)
    Holder<CourseScoreListType> courseScoreListTypeHolder)
    throws AuthorityException, IdNotFoundException, ScoreModifyException, ScoreTypeException
;

@Override
@Transactional
public void addScore(Holder<CourseScoreListType> courseScoreListTypeHolder)
    throws AuthorityException, IdNotFoundException, ScoreModifyException, ScoreTypeException {
    List<CourseScore> courseScores = courseScoreListTypeHolder.value.getCourseScores();
    validate(courseScores);
    courseScoreListTypeHolder.value.setCourseScores(courseScores.stream()
        .map(ScoreListEntity::new)
        .map(scoreListEntity -> scoreDao.save(scoreListEntity))
        .map(CourseScore::new).collect(Collectors.toList()));
}

```

(6) 配置 webservice

新建 config 类 (@Configuration) 注解, 添加以下配置：

```

@Bean
public ServletRegistrationBean dispatcherServlet() {
    return new ServletRegistrationBean(new CXFServlet(), ...urlMappings: "/soap-api/*");
}

@Bean(name= Bus.DEFAULT_BUS_ID)
public SpringBus springBus() { return new SpringBus(); }

```

随后注入刚刚写的 service

```

@Bean(name = "authEndpoint")
public Endpoint endpoint1() {
    EndpointImpl endpoint = new EndpointImpl(springBus(), authController);
    endpoint.publish( addr: "/auth");
    return endpoint;
}

@Bean(name = "scoreEndpoint")
public Endpoint endpoint2() {
    EndpointImpl endpoint = new EndpointImpl(springBus(), scoreController);
    endpoint.publish( addr: "/score");
    return endpoint;
}

@Bean(name = "studentEndpoint")
public Endpoint endpoint3() {
    EndpointImpl endpoint = new EndpointImpl(springBus(), studentController);
    endpoint.publish( addr: "/student");
    return endpoint;
}

```

(7) 运行项目

直接运行 main 方法即可

```

/**
 * Created by cuihao on 2017-06-11.
 * Main entrance of student manage application
 */
@SpringBootApplication
public class StudentApplication {

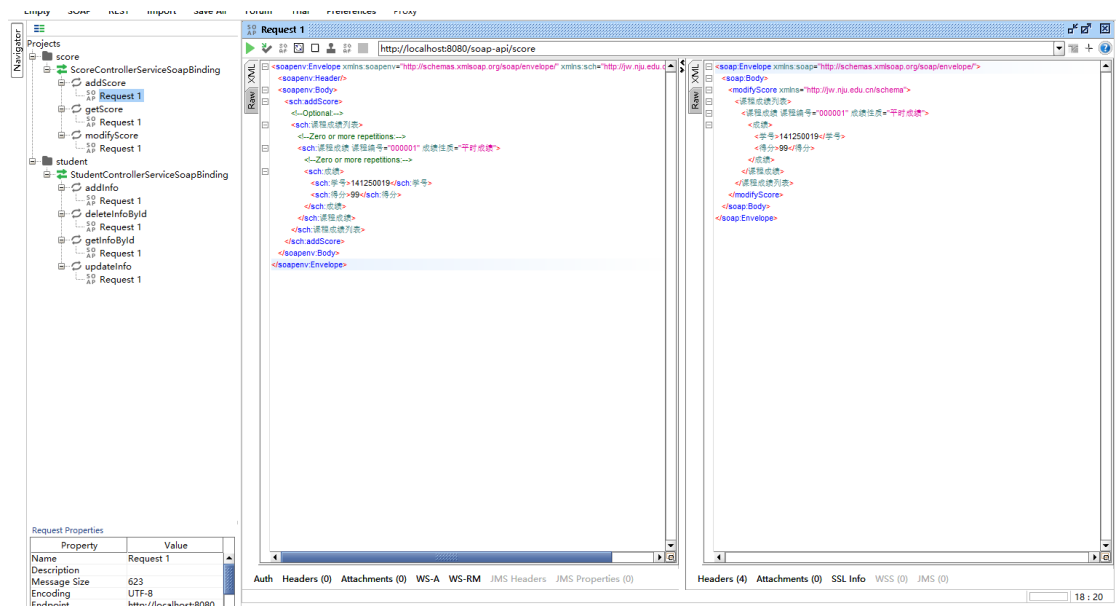
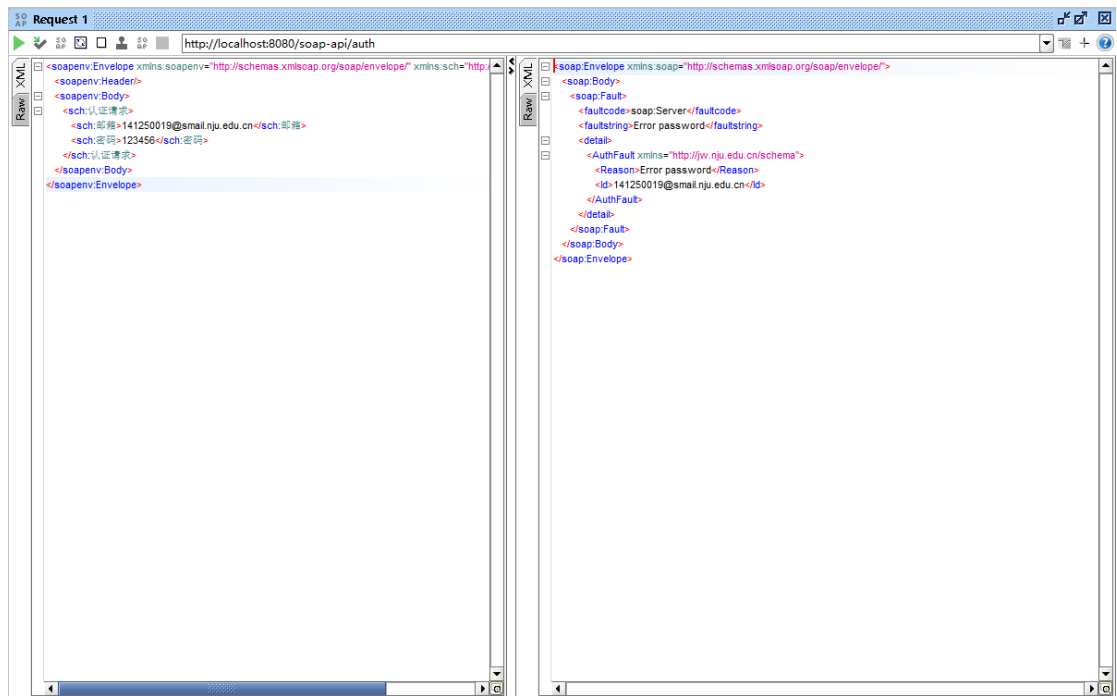
    public static void main(String[] args) { SpringApplication.run(StudentApplication.class, args); }
}

```

(8) 测试接口

使用 SoapUI 测试服务发布的接口：

测试 fault：



The screenshot displays the SoapUI interface with two XML messages. The left pane shows the request XML, and the right pane shows the response XML.

Request XML:

```

<?xml version='1.0' encoding='UTF-8'>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://jw.nj.edu.cn/schemas">
  <soap:Header/>
  <soap:Body>
    <sch:学号=141250019></sch:学号>
  </soap:Body>
</soap:Envelope>

```

Response XML:

```

<?xml version='1.0' encoding='UTF-8'>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <sch:成绩列表 xmlns="http://jw.nj.edu.cn/schemas">
      <成绩成绩 课程编号="000001">成绩性质="总评成绩">
        <成绩>
          <学号=141250019>学号</学号>
          <得分=100>得分</得分>
        </成绩>
      </成绩成绩>
      <成绩成绩 课程编号="000001">成绩性质="平时成绩">
        <成绩>
          <学号=141250019>学号</学号>
          <得分=67>得分</得分>
        </成绩>
      </成绩成绩>
    </sch:成绩列表>
  </soap:Body>
</soap:Envelope>

```

The response XML contains a table of exam results for two students. The first student (学号=141250019) has a total score (总评成绩) of 100 and a平时成绩 (平时成绩) of 67. The second student (学号=141250019) has a total score (总评成绩) of 100 and a平时成绩 (平时成绩) of 67.

The screenshot displays a web browser with two panels showing SOAP API interactions. The left panel shows a request to `http://localhost:8080/soap-api/score`, and the right panel shows the response.

Request (Left Panel):

```

<?xml version='1.0' encoding='utf-8'>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://www.rigit.edu.cn/schemas">
  <soap:Header/>
  <soap:Body>
    <sch:成绩成绩列表>
      <!--Zero or more repetitions-->
      <sch:成绩成绩 成绩编号="000001" 成绩性质="平时成绩">
        <!--Zero or more repetitions-->
        <sch:成绩>
          <sch:学号="141250019"/>
          <sch:得分="47"/>
        </sch:成绩>
      </sch:成绩>
    </sch:成绩成绩列表>
  </soap:Body>
</soap:Envelope>

```

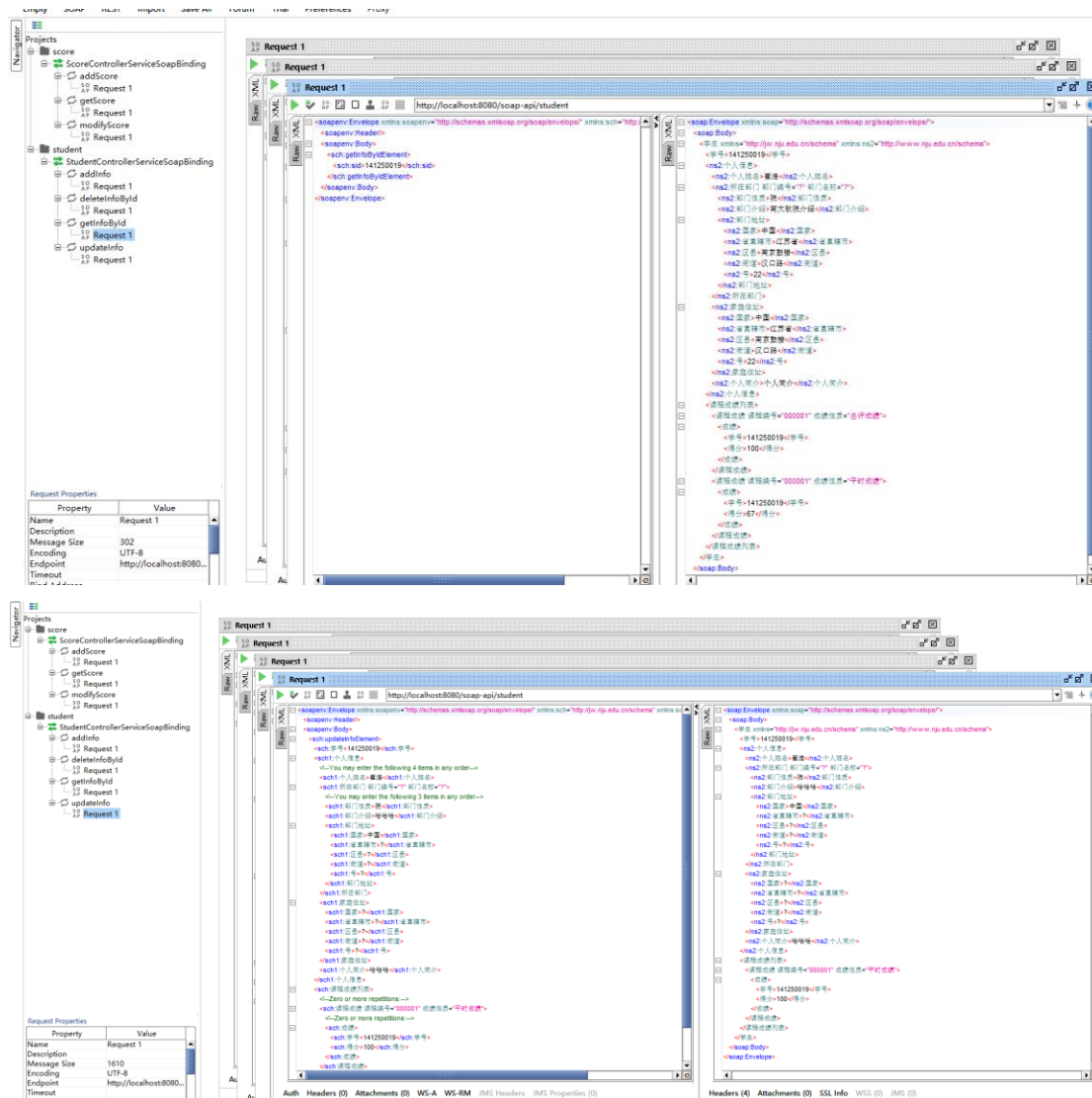
Response (Right Panel):

```

<?xml version='1.0' encoding='utf-8'>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <!--成绩成绩列表 xmlns="http://www.rigit.edu.cn/schemas">
      <成绩成绩 成绩编号="000001" 成绩性质="平时成绩">
        <成绩>
          <学号="141250019"/>
          <得分="47"/>
        </成绩>
      </成绩>
    </成绩成绩列表>
  </soap:Body>
</soap:Envelope>

```

The status bar at the bottom indicates the following tabs: Headers (4), Attachments (0), SSL Info, WSS (0), JMS (0).



(8) 发布服务

使用 `mvn package -Dmaven.test.skip=true` 生成 jar 包，在部署服务器上输入命令：
`nohup java -jar soa.jar &` 运行该 springboot 项目：

Available SOAP services:	
MyAuth <ul style="list-style-type: none"> verify 	Endpoint address: http://106.15.183.234:8080/soap-api/auth WSDL: http://jw.nju.edu.cn/schema/AuthControllerService Target namespace: http://jw.nju.edu.cn/schema
Score <ul style="list-style-type: none"> addScore getScore modifyScore 	Endpoint address: http://106.15.183.234:8080/soap-api/score WSDL: http://jw.nju.edu.cn/schema/ScoreControllerService Target namespace: http://jw.nju.edu.cn/schema
StudentPort <ul style="list-style-type: none"> deleteInfoById addInfo getInfoById updateInfo 	Endpoint address: http://106.15.183.234:8080/soap-api/student WSDL: http://jw.nju.edu.cn/schema/StudentControllerService Target namespace: http://jw.nju.edu.cn/schema

```
106.15.183.234:8080/soap-api/auth?wsdl
应用 ★ Bookmarks Learning Working Seeing Google Courses Login 博客频道 - CSDN!

This XML file does not appear to have any style information associated with it. The document tree is shown below.
<?xml:definition xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:wsa="http://www.w3.org/2006/05/addressing/wsdl" xmlns:wsoap="http://www.w3.org/2007/05/addressing/metadata"
xmlns:tns="http://jw.nju.edu.cn/schema" xmlns:soap="http://schemas.xmlsoap.org/soap/http" name="AuthControllerService" targetNamespace="http://jw.nju.edu.cn/schema">
  <wsdl:types>
    <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:tns="http://jw.nju.edu.cn/schema" attributeFormDefault="unqualified" elementFormDefault="qualified" targetNamespace="http://jw.nju.edu.cn/schema">
      <xsd:complexType name="账号认证类型">
        <xsd:sequence>
          <xsd:element name="邮箱" type="xsd:string"/>
          <xsd:element name="密码" type="xsd:string"/>
        </xsd:sequence>
      </xsd:complexType>
      <xsd:complexType name="验证类型">
        <xsd:sequence>
          <xsd:element name="学号" type="xsd:string"/>
          <xsd:element name="权限" type="tns:权限级别"/>
        </xsd:sequence>
      </xsd:complexType>
      <xsd:complexType name="NotFoundType">
        <xsd:sequence>
          <xsd:element name="NotFoundReason" type="tns:NotFoundReasonType"/>
          <xsd:element name="NotFoundId" type="xsd:string"/>
        </xsd:sequence>
      </xsd:complexType>
      <xsd:complexType name="AuthFaultType">
        <xsd:sequence>
          <xsd:element name="Reason" type="xsd:string"/>
          <xsd:element name="Id" type="xsd:string"/>
        </xsd:sequence>
      </xsd:complexType>
      <xsd:simpleType name="权限级别">
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="老师"/>
          <xsd:enumeration value="研究生"/>
          <xsd:enumeration value="本科生"/>
        </xsd:restriction>
      </xsd:simpleType>
      <xsd:simpleType name="NotFoundReasonType">
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="学号不存在"/>
          <xsd:enumeration value="未找到输入学号的成績"/>
          <xsd:enumeration value="课程不存在"/>
          <xsd:enumeration value="未找到输入课程的成績"/>
        </xsd:restriction>
      </xsd:simpleType>
      <xsd:element name="NotFoundFault" nillable="true" type="tns:NotFoundType"/>
      <xsd:element name="AuthFault" nillable="true" type="tns:AuthFaultType"/>
      <xsd:element name="认证请求" nillable="true" type="tns:账号认证类型"/>
    </xsd:schema>
  </wsdl:types>
  <wsdl:service name="AuthControllerService" targetNamespace="http://jw.nju.edu.cn/schema">
    <wsdl:operation name="认证" type="tns:账号认证类型"/>
    <wsdl:operation name="验证" type="tns:验证类型"/>
    <wsdl:operation name="认证请求" type="tns:账号认证类型"/>
  </wsdl:service>
</?xml:definition>
```

可以看到 start from java 自动生成的 wsdl 文件，其他两个服务的 wsdl 文件与原来的本地文件基本保持一致。