sb-webservice

基于spring boot 搭建web service服务。参加了一个旧项目的对接，该项目使用web service进行接口定义。web service本身是一个已经过时的技术，属于是JSP、Servlet向Restful API过渡时期的接口规范。

所以，对于web service来说，本身就是接口前身，理解web service就把它当成一种接口编程技术即可。

1. Web service相关概念

（1）web service介绍

Web Service是一个SOA（面向服务的编程）的架构，它是不依赖于语言，不依赖于平台，可以实现不同的语言间的相互调用，通过Internet进行基于Http协议的网络应用间的交互。（关键词：SOA、跨语言、跨平台、基于http协议）

Web Service要素：SOAP、WSDL

（2）SOAP(Simple Object Access Protocol)简单对象存取协议。是XML Web Service 的**通信协议**。当用户找到你的WSDL描述文档后，他通过可以SOAP调用你建立的Web服务中的一个或多个操作。SOAP是XML文档形式的调用方法的规范，它可以支持不同的底层接口。（关键词：通信协议、规范）SOAP在WSDL文件中定义portType时体现到。如下：

```
<wsdl:binding name="RoleServiceImplServiceSoapBinding" type="tns:StudentRoleService">
    <!--指定请求类型是xml 传输协议是soap-->
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <!--定义方法名、请求响应信息-->
    <wsdl:operation name="getUserRole">
      <soap:operation soapAction="" style="document"/>
      <wsdl:input name="getUserRole">
      </wsdl:input>
      <wsdl:output name="getUserRoleResponse">
      </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
```

（3）WSDL(Web Services Description Language) WSDL 文件是一个 XML 文档，用于说明一组 SOAP 消息以及如何交换这些消息。大多数情况下由软件自动生成和使用。（关键词：接口功能描述的XML文档）

2. Web service参考理解

通过对比Restful形式的接口，掌握其中的一些概念。假设现在需要编写一个用户服务，这个服务需要提供用户相关操作、提供角色相关操作、提供权限相关操作。

（1）WebService注解和RestController注解

以Restful接口来说，会定义UserController、RoleController、PermissionController，每个controller里面包含对应的功能方法，功能方法也就是接口对外提供的能力。**而web service则是定义service**，如下：

```java
// 类似于角色controller
@WebService(name = "StudentRoleService",
        targetNamespace = "http://service.ws.sb.ml/")
public interface RoleService {

    // 类似于角色controller提供的功能
    String getUserRole(String uID);
}
// 权限相关
@WebService(name = "StudentPermissionService",
        targetNamespace = "http://service.ws.sb.ml/")
public interface PermissionService {

    String geUserPermission(String uId);
}
// 用户相关
@WebService(name = "StudentService",
        targetNamespace = "http://service.ws.sb.ml/")
public interface UserService {
    @WebMethod
    boolean saveUser(User user);
    @WebMethod
    User getUserByID(String id);
    @WebMethod
    List<User> getAllUsers();
}
```

结论：@WebService注解的接口表示一个对外服务，类似于restful api的一个controller

（2）portType和service 层方法

portType是WebService WSDL中的一个标签，用于定义@WebService类声明的方法信息。

Restful API接下来会编写service层方法供controller层调用，并会在service层实现接口，但是对于web service而言则是直接实现接口。如下：

```java
@Slf4j
@Component
@WebService(name = "StudentPermissionService",
        targetNamespace = "http://service.ws.sb.ml/")
public class PermissionServiceImpl implements PermissionService {
    @Override
    public String geUserPermission(String uId) {
        log.info("请求uID: {}", uId);
        return "permission has [create, read]";
    }
}
```

小结：controller层使用的service层在web service中直接实现，无需定义业务类。

（3）其他

到这里就把核心概念梳理了。web service属于老旧的接口定义规范，相比现在spring boot开发restful api接口来说，很麻烦。麻烦点：

需要手动部署URL到实现类上

这一点就是说@WebService接口功能并没有@RestController接口那么强大，@RestController接口可以指定URL地址，并自动将请求转发到该实体类对象上，可是@WebService不行，需要通过**暴露点的形式做请求URL与该实体类对象的映射**。如下代码，就是将`/ws/api/role`的所有请求转发到roleService实例上。

⚠️⚠️这里有个非常重要的问题就是：如果如果WebService中定义了多个方法，就像UserService一样，但是只有一个URL，是如何做到区分调用哪个方法上的呢？答案就是SOAP规范，会在请求中指定调用的方法名。

⚠️⚠️**WSDL文件完整的接口URL路径：`http://127.0.0.1:8080/ws/api/role?wsdl`，该URL同时也是客户端请求路径，该URL形式万年不变，客户端请求必须请求这个URL地址。**

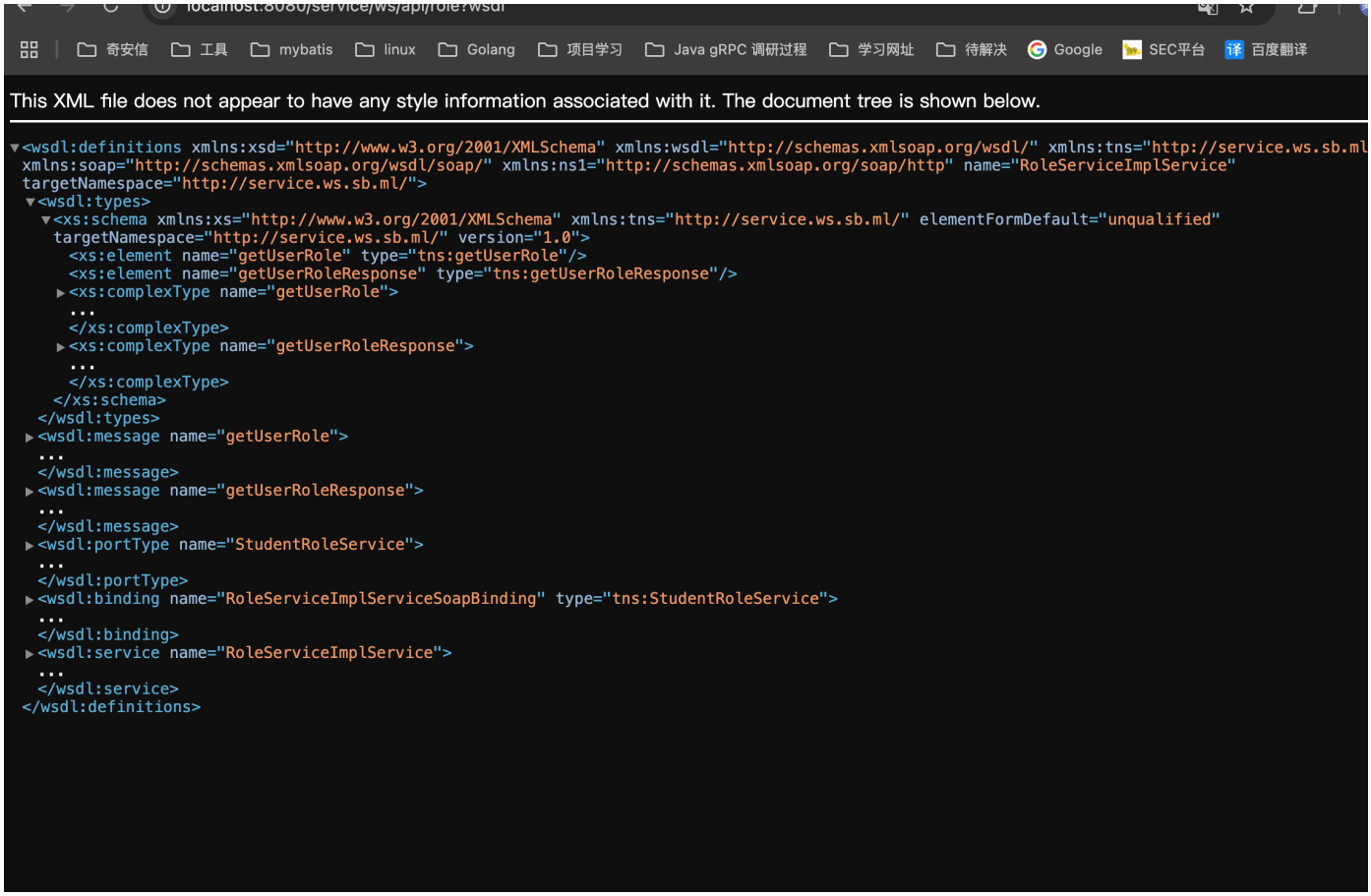
```
@Bean
public Endpoint roleEndPoint() {
  EndpointImpl endpoint = new EndpointImpl(springBus(), roleService);
  endpoint.publish("/ws/api/role");
  return endpoint;
}
```

（4）WSDL文件

如下是roleService WSDL文件截图。

- message标签定义了消息类型，web service一个请求产生两个消息：请求消息和响应消息。
- portType定义接口方法，包含名、请求消息和响应消息信息
- wsdl:binding定义接口方法请求方式、方法定义
- wsdl:service定义接口URL和实现类

总之一句话，web service在竭力利用该文件描述接口规范，接口规范包括了各个方面。



3. web service实战

基于`spring boot 2.7.13 + JDK 11`实现了一个web service服务端和客户端。代码就是本库代码，从实现的结果来看：

（1）web service应该可以和restful api共存，只要URL不冲突应该没事

（2）客户端请求如果可以拿到wsdl文件下，使用wsimport生成静态代理是一个不错的选择，但是就怕服务端升级代码，客户端也需要升级代码

（3）基于动态代理方式请求web service server可以避免生成静态代理代码，但是目前对接受复杂类型的返回结果，还未实现。