**Spring Boot + SOAP Web Service Simple Example**

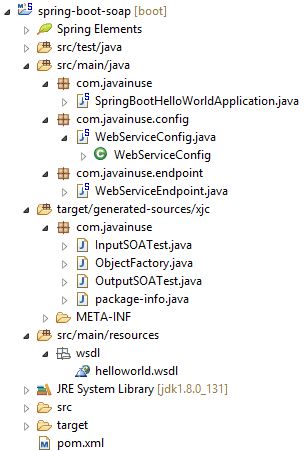
In this post we develop a Spring Boot Application to expose SOAP Webservices.

What is SOAP Webservice? Need for it?

Web Services can be implemented in either of the 2 ways-

* REST
* SOAP

A majority of the examples we have done before make use of the REST web services.  
SOAP (originally Simple Object Access Protocol) is a protocol specification for exchanging structured information in the implementation of web services in computer networks. SOAP allows processes running on disparate operating systems (such as Windows and Linux) to communicate using Extensible Markup Language (XML).  
SOAP can be used in conjunction with WSDL which is standardized what means that people who know the standard (WSDL) can learn from it what operations a web service offers and how data is exchanged.  
This knowledge can be used to create tools that generate type safe binder classes/objects out of the WSDL file.  
These generated classes (to make RPCs) can be used without needing to manually implementing the requests and encoding/parsing of the data that is exchanged.

  
The classes in package **com.javainuse** are generated classes from wsdl.  
The pom.xml will be as follows-

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.javainuse</groupId>

<artifactId>spring-boot-soap</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>SpringBootHelloWorld</name>

<description>Demo project for Spring Boot</description>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>1.4.1.RELEASE</version>

<relativePath /> <!-- lookup parent from repository -->

</parent>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>

<java.version>1.8</java.version>

<maven-jaxb2-plugin.version>0.13.2</maven-jaxb2-plugin.version>

</properties>

<dependencies>

**<dependency>**

**<groupId>org.springframework.boot</groupId>**

**<artifactId>spring-boot-starter-web-services</artifactId>**

**</dependency>**

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

<plugin>

<groupId>org.jvnet.jaxb2.maven2</groupId>

<artifactId>maven-jaxb2-plugin</artifactId>

<version>0</version>

**<executions>**

**<execution>**

**<goals>**

**<goal>generate</goal>**

**</goals>**

**</execution>**

**</executions>**

**<configuration>**

**<schemaDirectory>/src/main/resources/wsdl</schemaDirectory>**

**<schemaIncludes>**

**<include>\*.wsdl</include>**

**</schemaIncludes>**

**</configuration>**

</plugin>

</plugins>

</build>

</project>

We define a "contract-first" webservice using WSDL. In the contract-first web service, the "contract" i.e a WSDL definition of operations and endpoints and XML schema of the messages is created first, without actually writing any service code.

Next we define the wsdl file that takes a String as an input and returns another string as output.  
The WSDL is defined as follows-

<?xml version="1.0" encoding="ISO-8859-1"?>

<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

xmlns:tns="http://javainuse.com" xmlns:xs="http://www.w3.org/2001/XMLSchema"

xmlns:http="http://schemas.xmlsoap.org/wsdl/http/" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"

targetNamespace="http://javainuse.com">

<wsdl:types>

<xs:schema targetNamespace="http://javainuse.com">

<xs:element name="inputSOATest">

<xs:complexType>

<xs:sequence>

<xs:element type="xs:string" name="test" />

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:element name="outputSOATest">

<xs:complexType>

<xs:sequence>

<xs:element type="xs:string" name="result" />

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:schema>

</wsdl:types>

<!--Define input and output parameters -->

<wsdl:message name="inputSOATest">

<wsdl:part name="in" element="tns:inputSOATest" />

</wsdl:message>

<wsdl:message name="outputSOATest">

<wsdl:part name="out" element="tns:outputSOATest" />

</wsdl:message>

<!--Define port definition -->

<wsdl:portType name="SOATestEndpoint">

<wsdl:operation name="SOATest">

<wsdl:input message="tns:inputSOATest" />

<wsdl:output message="tns:outputSOATest" />

</wsdl:operation>

</wsdl:portType>

<!--Bind Soap operation and service -->

<wsdl:binding name="SOATestBinding" type="tns:SOATestEndpoint">

<soap:binding transport="http://schemas.xmlsoap.org/soap/http" />

<wsdl:operation name="SOATest">

<soap:operation soapAction="http://javainuse.com"

style="document" />

<wsdl:input>

<soap:body parts="in" use="literal" />

</wsdl:input>

<wsdl:output>

<soap:body parts="out" use="literal" />

</wsdl:output>

</wsdl:operation>

</wsdl:binding>

<!--Define service -->

<wsdl:service name="SOATestEndpointService">

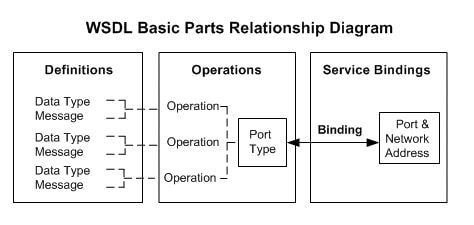
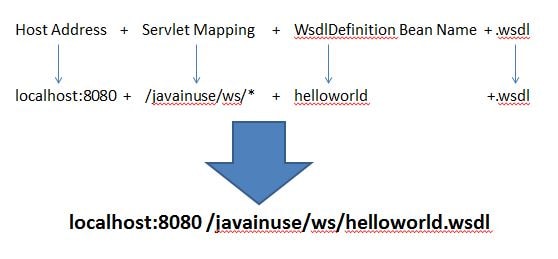
<wsdl:port name="SOATestEndpoint" binding="tns:SOATestBinding">

<soap:address location="http://localhost:8080/javainuse/ws/helloWorldExample" />

</wsdl:port>

</wsdl:service>

</wsdl:definitions>

  
Next using the ServletRegistrationBean we register the MessageDispatcherServlet with Spring Boot.  
During this registration the servlet mapping URI pattern is set to **/javainuse/ws/\***. Using this path, the web container will map incoming HTTP requests to the MessageDispatcherServlet. The DefaultWsdl11Definition exposes a standard WSDL 1.1 using the specified Hello World WSDL file. MessageDispatcherServlet also automatically detects any WsdlDefinition defined in its application context. So the path of the url at which WSDL is exposed will be as follows-  


package com.javainuse.config;

import org.springframework.boot.web.servlet.ServletRegistrationBean;

import org.springframework.context.ApplicationContext;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.core.io.ClassPathResource;

import org.springframework.ws.config.annotation.EnableWs;

import org.springframework.ws.config.annotation.WsConfigurerAdapter;

import org.springframework.ws.transport.http.MessageDispatcherServlet;

import org.springframework.ws.wsdl.wsdl11.SimpleWsdl11Definition;

import org.springframework.ws.wsdl.wsdl11.Wsdl11Definition;

@EnableWs

@Configuration

public class WebServiceConfig extends WsConfigurerAdapter {

@Bean

public ServletRegistrationBean messageDispatcherServlet(ApplicationContext applicationContext) {

MessageDispatcherServlet servlet = new MessageDispatcherServlet();

servlet.setApplicationContext(applicationContext);

return new ServletRegistrationBean(servlet, "/javainuse/ws/\*");

}

@Bean(name="helloworld")

public Wsdl11Definition defaultWsdl11Definition() {

SimpleWsdl11Definition wsdl11Definition = new SimpleWsdl11Definition();

wsdl11Definition.setWsdl(new ClassPathResource("/wsdl/helloworld.wsdl"));

return wsdl11Definition;

}

}

Next we define the endpoint which will process the incoming xml messages.

package com.javainuse.endpoint;

import org.springframework.ws.server.endpoint.annotation.Endpoint;

import org.springframework.ws.server.endpoint.annotation.PayloadRoot;

import org.springframework.ws.server.endpoint.annotation.RequestPayload;

import org.springframework.ws.server.endpoint.annotation.ResponsePayload;

import com.javainuse.InputSOATest;

import com.javainuse.ObjectFactory;

import com.javainuse.OutputSOATest;

**@Endpoint**

public class WebServiceEndpoint {

private static final String NAMESPACE\_URI = "http://javainuse.com";

@PayloadRoot(namespace = NAMESPACE\_URI, localPart = "inputSOATest")

@ResponsePayload

public OutputSOATest hello(@RequestPayload InputSOATest request) {

String outputString = "Hello " + request.getTest() + "!";

**ObjectFactory factory = new ObjectFactory();**

**OutputSOATest response = factory.createOutputSOATest();**

**response.setResult(outputString);**

return response;

}

}

Finally create the class with @SpringBootApplication annotation.

package com.javainuse;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringBootHelloWorldApplication {

public static void main(String[] args) {

SpringApplication.run(SpringBootHelloWorldApplication.class, args);

}

}

Start the application-  
  
  
Next go to **http://localhost:8080/javainuse/ws/helloworld.wsdl**  
  
Finally using the wsdl url create a SOAP UI project.  
