零壹激光 WebServer 使用说明书

目录

零壹激光 WebServer 使用说明书 1
1.简介 3
1.1 配置文件 3
1.2 协议选择 3
2. SOAP 协议 4
2.1 解析 4
3. JSON 协议 7
3.1 解析 7
4. 数据库 9
4.1 远程数据库 9
4.2 本地数据库 10

1.简介

零壹激光 WebServer 是为了解决激光板卡与网络服务器之间的通讯而编写的一个插件，适用于零壹软件的激光和气动系列软件。

主界面如下：

1.1 配置文件

设置好的配置可以保存成配置文件，可以导出到 U 盘然后导入到其他板卡上，方便批量设置。

1.2 协议选择

目前，WebServer 支持以下几种协议：

SOAP、JSON、数据库。

SOAP 又包括 1.1、1.2 和较老的 SOAP 版本，软件中称为 SOAP1。

数据库包括 SQLServer、Oracle、MySQL 和 SQLite，其中 SQLite 为本机数据库，需要导入使用。

2. SOAP 协议

SOAP 协议包括常见的 1.1、1.2 版本，以及比较老的 Axis1 的版本，这里称为 SOAP1。这里需要根据服务器的具体情况来设置。

```
<wsdl:port name="WQ01HttpSoap11Endpoint" binding="ns:WQ01Soap11Binding">
  <soap:address location="http://192.168.66.166:8080/axis2/services/WQ01.WQ01HttpSoap11Endpoint/"/>
</wsdl:port>
<wsdl:port name="WQ01HttpSoap12Endpoint" binding="ns:WQ01Soap12Binding">
  <soap12:address location="http://192.168.66.166:8080/axis2/services/WQ01.WQ01HttpSoap12Endpoint/"/>
</wsdl:port>
```

如上图，如果存在 Soap11Endpoint，则表示支持 1.1 版本，存在 Soap12Endpoint 则表示支持 1.2 版本。

```
</wsdl:binding>
<wsdl:service name="SayHelloService">
  <wsdl:port binding="impl:soaptestSoapBinding" name="soaptest">
    <wsdlsoap:address location="http://192.168.66.166:8080/axis/services/soaptest"/>
  </wsdl:port>
</wsdl:service>
```

如果是上图这样的 SoapBinding，则表示为 SOAP1 版本。

2.1 解析

这里以一个服务器实际例子做说明。

```
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:ns="http://service.weiqian01.com" xmlns:ax23="http://service.weiqian01.com/xsd" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ns1="http://org.apache.axis2/xsd" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/" xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/" xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsdl" targetNamespace="http://service.weiqian01.com">
  <wsdl:documentation>WQ01</wsdl:documentation>
  <wsdl:types>
    <xs:schema xmlns:ax24="http://service.weiqian01.com/xsd" attributeFormDefault="qualified" elementFormDefault="qualified" targetNamespace="http://service.weiqian01.com">
      <xs:import namespace="http://service.weiqian01.com/xsd"/>
      <xs:element name="getData">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" name="id" nillable="true" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="getDataResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" name="return" nillable="true" type="ax24:bluetooth"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>

    </xs:schema>
    <xs:schema attributeFormDefault="qualified" elementFormDefault="qualified" targetNamespace="http://service.weiqian01.com/xsd">
      <xs:complexType name="bluetooth">
        <xs:sequence>
          <xs:element minOccurs="0" name="jianiu" nillable="true" type="xs:string"/>
          <xs:element minOccurs="0" name="sn" nillable="true" type="xs:string"/>
        </xs:sequence>
      </xs:complexType>
    </xs:schema>
```

这是部署在本地服务器上的一个例子，在本例中，完整的服务访问地址是 http://192.168.66.166:8080/axis2/services/WQ01?wsdl ，命名地址是 http://service.weiqian01.com，获取数据的方法名称是 getData，参数为 id，类型为字符串；返回数据的方法名称是 getDataResponse，参数为 return，类型为自定义的 bluetooth 类，即包含两个参数 jianiu 和 sn，均为字符串。重点信息在上图中以红框标出，根据上面的信息可以填写下图配置：

发送设置：即从文档内容映射到参数内容，本例中按下图填写：

表示文档中标记名为 A 的标记内容，会填写到参数名为 id 的字段中，在执行的时候自动替换。

接收设置：即从接收到的 xml 文档中解析出相关参数，然后映射到文档内容中，本例中按下图填写：

表示参数“jianiu”中的实际值会自动替换标记名为 B 的标记内容，参数“sn”中的实际值会自动替换标记名为 C 的标记内容。

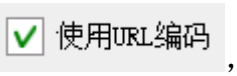
参考 SoapUI 软件中实际执行效果：

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ser="http://service.weiqian01.com">
  <soapenv:Header/>
  <soapenv:Body>
    <ser:getData>
      <!--Optional:-->
      <ser:id>001122</ser:id>
    </ser:getData>
  </soapenv:Body>
</soapenv:Envelope>
```

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <ns:getDataResponse xmlns:ns="http://service.weiqian01.com">
      <ns:return xsi:type="ax23:bluetooth" xmlns:ax23="http://service.weiqian01.com/xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <ax23:jianiu>01</ax23:jianiu>
        <ax23:sn>010203</ax23:sn>
      </ns:return>
    </ns:getDataResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

3. JSON 协议

JSON 协议支持 Get 和 Post 两种方式。根据服务器实际情况设置。

默认情况下，http 请求中 Content-Type 为 application/json，如果服务器使用的是较老的方案，可以勾选使用 URL 编码 

，这样 Content-Type 会修改为 application/x-www-form-urlencoded。

3.1 解析

这里以一个服务器实际例子做说明。

这里通过 Get 的方式获取数据。完整的查询 URL 为 http://192.168.66.166:8666/api/getImeiInfo?ProjectName=A01&SNOrIMEI=imei，这里只需要填写前面固定的部分，即 http://192.168.66.166:8666/api/getImeiInfo，问号后面的为查询部分，放在发送设置内填写。

消息映射中，左边依旧是选择文档中的标记名，与 SOAP 的发送设置略有区别，右边的参数名需要直接填写 URL 中的参数名，如例子中的 ProjectName 和 SNOrIMEI，程序会自动生成完整的查询 URL。

接收设置与 SOAP 一致。

JSON 还支持使用脚本的方式进行查询。如下面的例子：

依旧是同一个接口，这里用 POST 的方式发送脚本内容。

可以看到在发送和接收脚本区域，都是直接使用 *“@标记名@”* 来替换标记内容，发送端为标记内容替换到脚本，接收端为脚本内容替换到标记。这里的脚本需要根据实际的发送接收的脚本来填写，避免服务器无法识别。

JSON 脚本也支持数组，在脚本和地址中使用数组需要用中括号（[]）将标记名扩起来，如下图：

```
http://192.168.66.166:8666/api/values?value={"pinbraodid":"@pinbraodid@","listpcbsn":["@pcbsn@"]}
```

这里的 pcbsn 就是一个数组。

数组内容的映射需要在文档中标记名体现，这里就需要标记名修改为 pcbsn[1]、pcbsn[2]….psbsn[n]的形式表示数组。

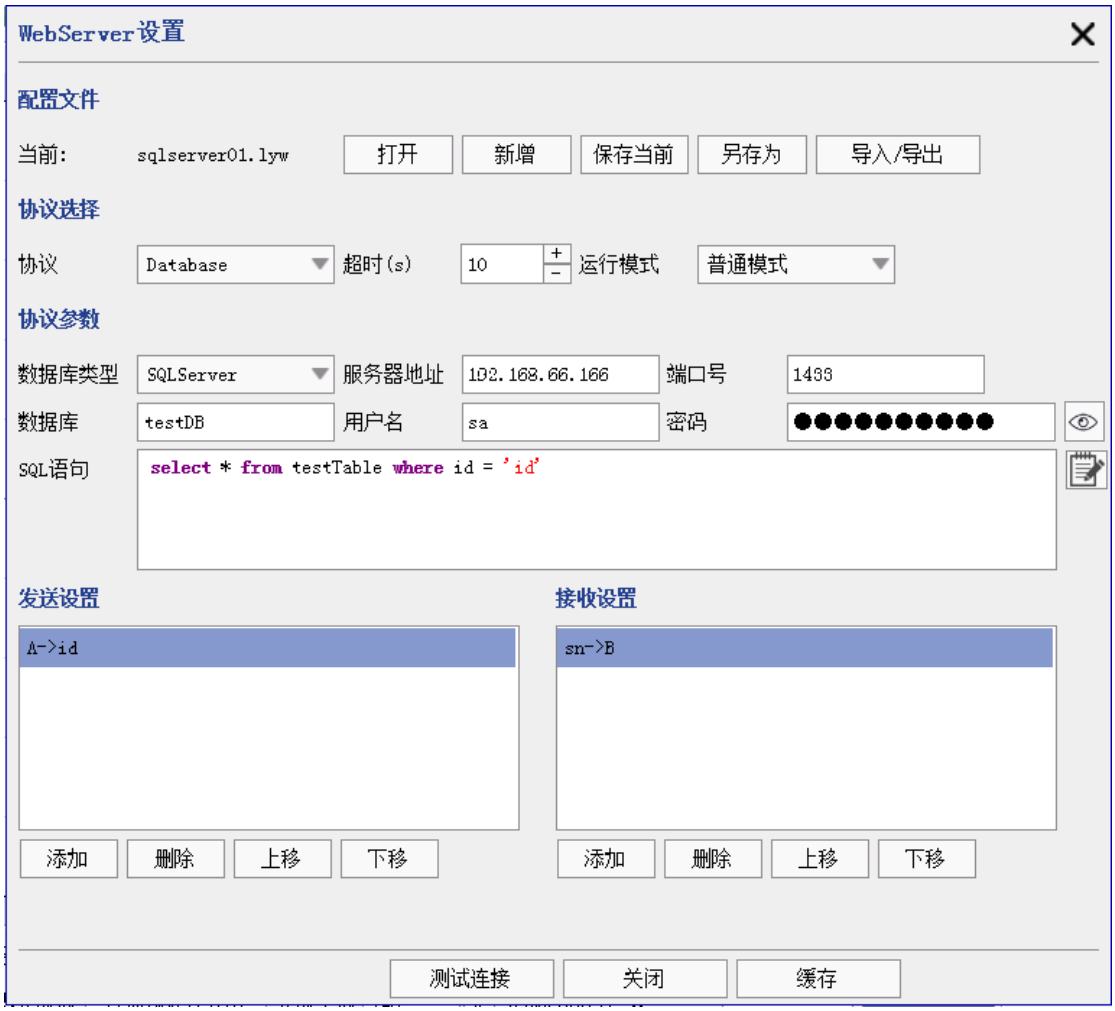
4. 数据库

数据库目前支持 SQLServer，Oracle，MySQL 以及本地数据库 SQLite 的增查删改的常用功能。

4.1 远程数据库

远程数据库指 SQLServer、Oracle 和 MySQL 这些需要网络连接到远程服务器获取数据的数据库。

以 SQLServer 为例：



服务器参数等按照实际情况填写。

SQL 语句填写查询语句，这里如果有查询条件，则把参数用单引号”包含起来，如上图中的 id。然后在发送设置中设置标记 A 的内容传送给参数 id。最终形成完整的 SQL 语句。如标记 A 的内容为 000001，那么最后形成的 SQL 语句就是 select * from testTable where id = ‘000001’。

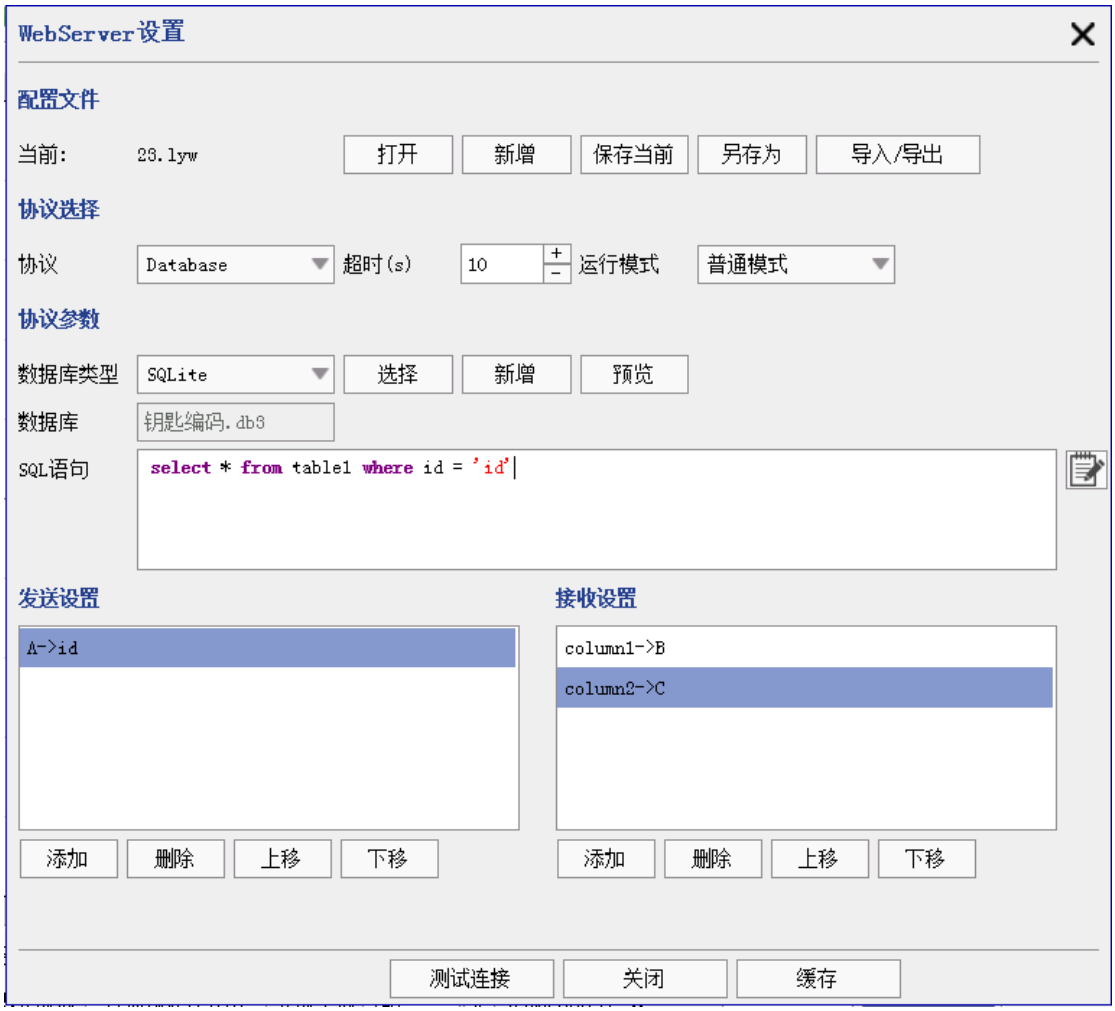
接收设置与 SOAP 类似。

MySQL 和 Oracle 的使用和 SQLServer 基本一致，Oracle 不同之处在于数据库这栏需要填写服务名（默认 orcl）。

4.2 本地数据库

本地数据库指的是 SQLite3 数据库，这个数据库是文件型数据库，只支持本地访问。

如下图：



可以选择一个从外部导入的 SQLite3 数据库文件，后缀名需为 db3。或者通过资源管理/升级功能从 U 盘导入 Excel 文件等生成数据库文件。选择文件后可以点击预览按钮查看数据库是否是需要的内容。

其余用法与 SQLServer 一致。