SoapGen User Guide

# Introduction

SoapGen is a simplified SOAP request message generator built on soapUI 4.5.0 compiled sources. This class was created with the intention of simplifying the generation and population of SOAP message fields directly from a WSDL at runtime without the need for an intermediate java code to be compiled and run. Using the soapUI classes directly will be tedious to understand and execute as the usage methods are not very well documented. In addition, it uses a mix of its own developed classes to read WSDLs and Apache's xmlbeans classes to parse schemas. SoapGen attempts to consolidate and create a set of simple APIs that is user friendly to easily parse a WSDL and generate its request messages.

# Concepts

## SoapGen's Message Field Map

Message fields within SoapGen are stored within HashMaps nested 2 levels down. Level 0 represents the interface name to operation HashMap mapping, level 1 represents the operation name to message fields HashMap mapping and level 2 represents the actual mapping of the message fields to the data itself. The nested HashMaps are generated upon instantiating the SoapGen object.



Figure Diagram representation of message field map

## Populating a SOAP Request Message

The method in which SoapGen populates the SOAP request message is by using the soapUI classes and methods to generate out the blank template request message, following which it is reparsed using an XML parser found within the soapUI. The data is then populated into the relavent fields via XPATH commands and then rendered back into the string.

The XPATH commands are generated using the message field names in the Message Field Map.

# Generating a SOAP Request using SoapGen

1. Create a new SoapGen object passing the path of the WSDL as the constructor's parameter.
2. You can get the interfaces and operations and its message fields by using getInterfaceNames(), getOperationNames() and getMessageFieldNames().
3. With the correct names in memory, you can set the fields required by using the setMessageField() method. Note that if you are passing XML data in or embedding a SOAP message within a SOAP message, declare true in the isXml parameter; this will append the CDATA tag. *Note: Message fields do not have to be populated in order.*
4. After you are done setting the fields, call generateSoapRequest() to build the SOAP request message.
5. If you need to retrieve destination information, use getEndpoints() and getSoapAction().

Included is SoapGenUI a CLI java application that is used for testing and development. However the app serves as a useful demo of the functionalities of SoapGen and the source code as a reference implementation. It is important to note that when using SoapGen APIs in a runtime environment, it is assumed that the developer would already know the WSDL, interface, operation and message field names to be populated either via hardcode or retrieved dynamically from an external source. When setting message fields using the setMessageField() method and passing in dynamically retrieved interface, operation and message field names, please check that the method returns a true after every call as the method does not throw an exception. As some message field names may not be unique, the names are stored as a path to ensure that the correct fields are populated correctly, as such, all field names include the full path from the SOAP body. In doubt, please refer to the SoapGenUI app for reference.

# Known Issues

* Functionality to parse and populated attributes in XML elements is not implemented. All attributes are ignored without exceptions or notifications.
* Input types are not checked and validated, meaning that you can pass strings into integer only fields.