Proxy Server Settings

In this lesson, we will learn how to add a proxy setting to the SOAP web service client.

We'll cover the following

- What is a proxy server?
- Adding a proxy setting to WebServiceTemplate

What is a proxy server?

A proxy server is an intermediary server that is virtually located between the client and the server. All the service requests are sent to the actual server via proxy servers.

Adding a proxy setting to `WebServiceTemplate`

We have already learned in the ***Sending requests using SOAP client*** lesson how to create a `WebServiceTemplate` that can be used to make web service calls.

This lesson is an extension of that part and here, we will add proxy settings to the created `WebServiceTemplate` with the following code:

```
import javax.annotation.PostConstruct;

import org.apache.http.HttpHost;
import org.apache.http.client.config.RequestConfig;
import org.apache.http.impl.client.CloseableHttpClient;
import org.apache.http.impl.client.HttpClients;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.oxm.jaxb.Jaxb2Marshaller;
import org.springframework.util.ClassUtils;
import org.springframework.ws.client.core.WebServiceTemplate;
import org.springframework.ws.transport.WebServiceMessageSender;
import org.springframework.ws.transport.http.HttpComponentsMessageSender;

import io.educative.soap_automation.GetStudentsRequest;

@Configuration
```

```
@Configuration
public class WebServiceClient {

    private final Jaxb2Marshaller marshaller = new Jaxb2Marshaller();

    private final static String PROXY_SERVER = "proxyhost";

    @PostConstruct
    public void init() throws Exception {
        marshaller.setPackagesToScan(ClassUtils.getPackageName(GetStudentsRequ
est.class));
        marshaller.afterPropertiesSet();
    }

    @Bean
    public WebServiceTemplate getWebServiceTemplate() {
        WebServiceTemplate webServiceTemplate = new WebServiceTemplate(marshal
ler);
        webServiceTemplate.setMessageSender(getMessageSender());
        return webServiceTemplate;
    }

    private WebServiceMessageSender getMessageSender() {
        RequestConfig config = RequestConfig
                .custom()
                .setProxy(new HttpHost(PROXY_SERVER))
                .build();

        CloseableHttpClient client = HttpClients
                .custom()
                .setDefaultRequestConfig(config)
                .build();

        return new HttpComponentsMessageSender(client);

    }
}
```

In the example above, we create an instance of Apache `HttpClient` to which we add the proxy settings, since we are considering sending all the requests to the web service over HTTP.

This instance of Apache `HttpClient` is used to create an instance of `HttpComponentsMessageSender` which is used to interact with the web service over `HTTP`. This instance is set to `WebServiceTemplate` as seen in the code snippet below:

```
WebServiceTemplate webServiceTemplate = new WebServiceTemplate(marshaller);
webServiceTemplate.setMessageSender(getMessageSender());
```

In this lesson, we learned how to create the `WebServiceTemplate` with proxy settings. Let us quickly take a quiz to see how much you understand about SOAP automation.