VVA OLA-Webservice / Online-Lieferbarkeits-Abfrage

Version 2.1 Technische Erläuterungen

Adresse und Definition des Webservice

Die neue Adresse des OLA-Webservice 2.0 ist:
https://www.vva-opus.de:443/ola-webservice/service/ola

Hier liegt auch die URL der Webservicebeschreibung:
https://www.vva-opus.de/ola-webservice/service/ola.wsdl

(WSDL = Webservice Definiton Language)

In der WSDL sind kleinere Unterschiede in der XML-Schema-Definition (XSD) zur Version 1.0, im Wesentlichen sind das folgende Korrekturen:

EinzelTitelRequest statt bisher einzelTitel

EinzelTitelResponse statt bisher einzelTitelResponse

AuftragRequest statt bisher auftrag

AuftragResponse statt bisher auftragResponse

Generierung des Client-Codes

Mit einem 8er-JDK kann z.B. der Client-Code für den Zugriff auf den Webservice mit dem Tool wsimport generiert werden. Wsimport liegt im bin-Verzeichnis des JDKs.

wsimport -keep -p <targetpackage> https://www.vva-opus.de/ola-webservice/service/ola.wsdl

Den Client kann man auch beispielsweise mit Hilfe von Spring Boot (Teil des Springframeworks) entwickeln.

https://howtodoinjava.com/spring-boot/spring-soap-client-webservicetemplate/

https://spring.io/guides/gs/consuming-web-service/

Authentifizierung

Jeder Benutzer erhält einen von der zentralen Handelsbetreuung vergebenen Benutzernamen und ein Passwort.

Zur Authentifizierung wird beim OLA-Webservice die HTTP-Header Basic Authentication verwendet.

In jedem Request muss im HTTP-Header der Benutzername und das Passwort mitgegeben werden.

Die Information „Benutzername:Passwort“ wird Base64-codiert und zusammen mit dem Präfix „Authorization: Basic“ dem HTTP-Header hinzugefügt.

Beispiel:

Benutzername = foo / Passwort = bar

Authorization: Basic Zm9vOmJhcg==

Beispielcode: Java-Methode, die aus Benutzername und Passwort den codierten String liefert:

```
import java.util.Base64;
...
private static String getEncodedAuthString(String user, String pw) {

        String userAuth = user + ":" + pw;
        String encodedUserAuth = Base64.getEncoder().encodeToString(userAuth.getBytes());

        return "Basic " + encodedUserAuth;
}
```

Achtung: Base64-Kodierung ist nur eine technische Kodierung von User:Passwort, jedoch keine Verschlüsselung. Da der VVA-OLA-Webservice nur über HTTPS erreichbar ist, wird über SSL/TLS die verschlüsselte Übertragung sichergestellt.

Mit dem Kommandozeilen-Programm curl (Client for URLs / gibt es unter Linux und Windows) kann man sich den Authorization-String ebenfalls generieren lassen.

```
curl -vL --user foo:bar https://www.vva-opus.de/ola-webservice/service/ola.wsdl
```

Mit curl kann man dann auch direkt den Webservice konsumieren bzw. testen. Im folgenden Beispiel wird der OLA-Webservice aufgerufen. Hinter –data wird die Datei mit dem Testrequest mitgegeben. Weiterhin sieht man, dass im Header der Base64-codierte Authorization-String steht.

```
curl -H "Authorization: Basic Zm9vOmJhcg==" --header "Content-Type:
text/xml;charset=UTF-8" --data @OLA-Testrequest.soap https://www.vva-opus.de/ola-
webservice/service/ola
```

Beispiele: Datei OLA-Testrequest.soap

a) Einzeltitel-Request

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
    <Body>
        <EinzelTitelRequest xmlns="http://ola">
            <positionsnummer></positionsnummer>
            <artikelnummer>123/04711</artikelnummer>
        </EinzelTitelRequest>
    </Body>
</Envelope>
```

b) Auftrag-Request (hier 3 Positionen)

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
    <Body>
        <AuftragRequest xmlns="http://ola">
            <EinzelTitelRequest>
                <positionsnummer>1</positionsnummer>
                <artikelnummer>123/04711</artikelnummer>
            </EinzelTitelRequest>
            <EinzelTitelRequest>
                <positionsnummer>2</positionsnummer>
                <artikelnummer>123/04712</artikelnummer>
            </EinzelTitelRequest>
            <EinzelTitelRequest>
                <positionsnummer>3</positionsnummer>
                <artikelnummer>123/04713</artikelnummer>
            </EinzelTitelRequest>
        </AuftragRequest>
    </Body>
</Envelope>
```

c) Einzeltitel-Request (ohne <positionsnummer>)

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
    <Body>
        <EinzelTitelRequest xmlns="http://ola">
            <artikelnummer>123/04711</artikelnummer>
        </EinzelTitelRequest>
    </Body>
</Envelope>
```

Anmerkung:
Das Feld <positionsnummer> ist wie in der WSDL beschrieben optional (minOccurs=0). Wenn es nicht im Request enthalten ist, so wird auch im Response in der Lieferbarkeitsmeldung die Positionsnummer weggelassen.

Beispiel-Response (Einzeltitel-Request ohne Positionsnummer)

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Header/>
<SOAP-ENV:Body>
   <ns2:EinzelTitelResponse xmlns:ns2="http://ola">
      <ns2:retTitel>
         <ns2:artikelnummer>123/04711</ns2:artikelnummer>
         <ns2:artikelbezeichnung>Beispiel</ns2:artikelbezeichnung>
         <ns2:verlagsname>Beispiel-Verlag</ns2:verlagsname>
         <ns2:lieferbarkeit>JA</ns2:lieferbarkeit>
         <ns2:meldetext>Lieferbar innerhalb … </ns2:meldetext>
      </ns2:retTitel>
   </ns2:EinzelTitelResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Wenn es interne Fehler gibt, so wird bei beiden Webservice-Methoden ein leerer Response zurückgegen.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Header/>
<SOAP-ENV:Body>
   <ns2:EinzelTitelResponse xmlns:ns2=http://ola/>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Im Fehlerfall loggen wir den Fehler bei uns intern serverseitig und geben den StackTrace nicht im SOAP-Response zurück.