



LinguLab API

Webservice Version 1.4
Description and Developer Guide

06.06.2011



Table of contents

1	Introduction	5
1.1	Web Service Overview.....	5
1.2	Web Service concept	6
1.3	Web Service Architecture.....	16
2	Login web-method	18
2.1	Overview.....	18
2.2	Error codes	18
2.3	Request and response.....	18
2.4	Using with .NET Framework 2.0	19
3	ValidateAuthentication web-method	20
3.1	Overview.....	20
3.2	Error codes	20
3.3	Request and response.....	20
	Using with .NET Framework 2.0	21
4	GetLanguages web-method	22
4.1	Overview.....	22
4.2	Error codes	22
4.3	Request and response	22
	Using with .NET Framework 2.0	23
5	GetConfigurations web-method	24
5.1	Overview.....	24
5.2	Error codes	24
5.3	Processing configurations	24
5.4	Request and response	26
5.5	Using with .NET Framework 2.0	27
6	ProcessText web-method	28
6.1	Overview.....	28
6.2	Processing text format	28
6.3	Error codes	29
6.4	Request and response	29
6.5	Using with .NET Framework 2.0	30
7	GetUpdatedText web-method	32
7.1	Overview.....	32
7.2	Error codes	33



7.3	Request and response	33
7.4	Using with .NET Framework 2.0	34
8	Example Implementations.....	35
8.1	ASP.NET 2.0	35
8.2	PHP	45
8.2.1	Overview.....	45
8.2.2	Example.....	45

1 Changelog

Version	Date	Changes
1.4	20.04.2011	Added Method GetLanguages() Returns List of available Languages which can be used for processing text

2 Introduction

This developer guide describes all steps and requirements to integrate the LinguLab services to your application and/or website. The web service provides an authenticated and secure access to the LinguLab Software. Your users can use the LinguLab Readability Index to measure and improve the quality of their written text.

The web service returns an image with the star rating and a link to the analysis details.

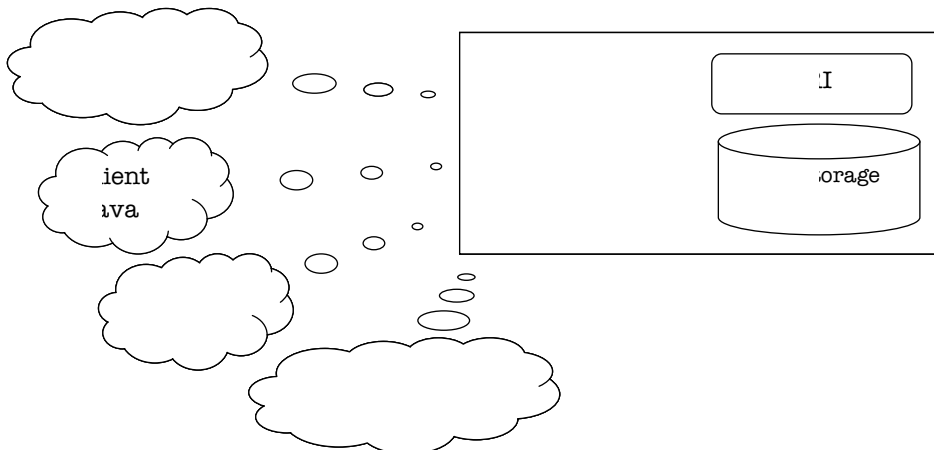
In the next chapter we show you the architecture of the system and how you can process text using our web service.

2.1 Web Service Overview

Analyzing a text requires two roles: a Client (your application) and our web service. For the client, you can use any programming language, like

- PHP,
- Java,
- Visual Basic,
- C#,
- Ruby on Rails.

The client communicates with the web service using the protocol SOAP. LinguLab supports version 1.1 and 1.2 of the SOAP protocol.

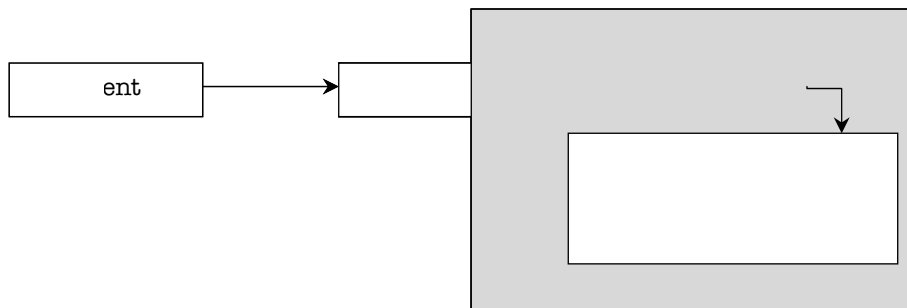


2.2 Web Service concept

Web service for processing text is created using SOAP.

SOAP, Simple Object Access Protocol, is a protocol specification for exchanging structured information in the implementation of web services in computer networks. It relies on Extensible Markup Language (XML) as its message format and usually relies on other Application Layer protocols, most notably Remote Procedure Call (RPC) and HTTP for message negotiation and transmission. SOAP can form the foundation layer of a web services protocol stack, providing a basic messaging framework upon which web services can be built.

All interaction with Web Service API can be made using HTTP Post, Soap 1.1 and 1.2.



Web service works through SOAP protocol and has WSDL scheme. To get scheme for Web Service go to URL

“<http://www.lingulab.net/Service/LiveService.asmx?wsdl>”:

```
<?xml version="1.0" encoding="utf-8" ?>
<wSDL:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:tns="https://api.lingulab.net/LiveService.asmx"
xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
targetNamespace="https://api.lingulab.net/LiveService.asmx"
xmlns:wSDL="http://schemas.xmlsoap.org/wsdl/">
<wSDL:types>
<s:schema elementFormDefault="qualified"
targetNamespace="https://api.lingulab.net/LiveService.asmx">
<s:element name="Login">
<s:complexType>
<s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="userName" type="s:string" />
<s:element minOccurs="0" maxOccurs="1" name="password" type="s:string" />
<s:element minOccurs="0" maxOccurs="1" name="pluginId" type="s:string" />
```

```
</s:sequence>
</s:complexType>
</s:element>
- <s:element name="LoginResponse">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="LoginResult" type="tns:LoginResult" />
</s:sequence>
</s:complexType>
</s:element>
- <s:complexType name="LoginResult">
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="AuthenticationKey" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="FullName" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="RoleName" type="s:string" />
  <s:element minOccurs="1" maxOccurs="1" name="ErrorCode" type="s:int" />
  <s:element minOccurs="0" maxOccurs="1" name="ErrorMessage" type="s:string" />
</s:sequence>
</s:complexType>
- <s:element name="ProcessText">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="inputData" type="tns:ProcessTextData" />
  <s:element minOccurs="0" maxOccurs="1" name="authenticationKey" type="s:string" />
</s:sequence>
</s:complexType>
</s:element>
- <s:complexType name="ProcessTextData">
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="Text" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="ConfigurationId" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="LanguageKey" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="SearchKeyword1" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="SearchKeyword2" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="SearchKeyword3" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="SectionTypeSettings"
type="tns:SectionTypeSetting" />
</s:sequence>
</s:complexType>
- <s:complexType name="SectionTypeSetting">
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="Headline" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="SubHeadline" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="Teaser" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="Text" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="BulletList" type="tns:ListSetting" />
  <s:element minOccurs="0" maxOccurs="1" name="NumberList" type="tns:ListSetting" />
</s:sequence>
```

```
</s:complexType>
- <s:complexType name="ListSetting">
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="ListTag" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="ItemTag" type="s:string" />
</s:sequence>
</s:complexType>
- <s:element name="ProcessTextResponse">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="ProcessTextResult"
type="tns:ProcessTextResult" />
</s:sequence>
</s:complexType>
</s:element>
- <s:complexType name="ProcessTextResult">
- <s:sequence>
  <s:element minOccurs="1" maxOccurs="1" name="ResultId" type="s:int" />
  <s:element minOccurs="1" maxOccurs="1" name="Measure" type="s:int" />
  <s:element minOccurs="0" maxOccurs="1" name="MeasureStarsHtml" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="LinkOnResultPage" type="s:string" />
  <s:element minOccurs="1" maxOccurs="1" name="ErrorCode" type="s:int" />
  <s:element minOccurs="0" maxOccurs="1" name="ErrorMessage" type="s:string" />
</s:sequence>
</s:complexType>
- <s:element name="ProcessTextForOffice">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="inputData" type="tns:ProcessTextData" />
  <s:element minOccurs="0" maxOccurs="1" name="authenticationKey" type="s:string" />
</s:sequence>
</s:complexType>
</s:element>
- <s:element name="ProcessTextForOfficeResponse">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="ProcessTextForOfficeResult"
type="tns:ProcessTextResult" />
</s:sequence>
</s:complexType>
</s:element>
- <s:element name="GetUpdatedText">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="1" maxOccurs="1" name="resultKey" type="s:int" />
  <s:element minOccurs="0" maxOccurs="1" name="authenticationKey" type="s:string" />
</s:sequence>
</s:complexType>
```

```
</s:element>
- <s:element name="GetUpdatedTextResponse">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="GetUpdatedTextResult"
type="tns:UpdatedTextResult" />
</s:sequence>
</s:complexType>
</s:element>
- <s:complexType name="UpdatedTextResult">
- <s:sequence>
  <s:element minOccurs="1" maxOccurs="1" name="ResultId" type="s:int" />
  <s:element minOccurs="1" maxOccurs="1" name="Measure" type="s:int" />
  <s:element minOccurs="0" maxOccurs="1" name="MeasureStarsHtml" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="LinkOnResultPage" type="s:string" />
  <s:element minOccurs="1" maxOccurs="1" name="ErrorCode" type="s:int" />
  <s:element minOccurs="0" maxOccurs="1" name="ErrorMessage" type="s:string" />
  <s:element minOccurs="1" maxOccurs="1" name="Version" type="s:int" />
  <s:element minOccurs="0" maxOccurs="1" name="RawText" type="s:string" />
</s:sequence>
</s:complexType>
- <s:element name="GetConfigurations">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="authenticationKey" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="languageKey" type="s:string" />
</s:sequence>
</s:complexType>
</s:element>
- <s:element name="GetConfigurationsResponse">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="GetConfigurationsResult"
type="tns:ConfigurationResult" />
</s:sequence>
</s:complexType>
</s:element>
- <s:complexType name="ConfigurationResult">
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="Configurations"
type="tns:ArrayOfConfigurationEntry" />
  <s:element minOccurs="1" maxOccurs="1" name="ErrorCode" type="s:int" />
  <s:element minOccurs="0" maxOccurs="1" name="ErrorMessage" type="s:string" />
</s:sequence>
</s:complexType>
- <s:complexType name="ArrayOfConfigurationEntry">
- <s:sequence>
```

```
<s:element minOccurs="0" maxOccurs="unbounded" name="ConfigurationEntry" nillable="true"
type="tns:ConfigurationEntry" />
</s:sequence>
</s:complexType>
<!-- <s:complexType name="ConfigurationEntry">
<!-- <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="Id" type="s:string" />
<s:element minOccurs="0" maxOccurs="1" name="Name" type="s:string" />
<s:element minOccurs="1" maxOccurs="1" name="IsKeywordsSupported" type="s:boolean" />
</s:sequence>
</s:complexType>
<!-- <s:element name="ValidateAuthentication">
<!-- <s:complexType>
<!-- <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="authenticationKey" type="s:string" />
</s:sequence>
</s:complexType>
</s:element>
<!-- <s:element name="ValidateAuthenticationResponse">
<!-- <s:complexType>
<!-- <s:sequence>
<s:element minOccurs="1" maxOccurs="1" name="ValidateAuthenticationResult" type="s:int"
/>
</s:sequence>
</s:complexType>
</s:element>
<!-- <s:element name="GetLanguages">
<!-- <s:complexType>
<!-- <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="authenticationKey" type="s:string" />
</s:sequence>
</s:complexType>
</s:element>
<!-- <s:element name="GetLanguagesResponse">
<!-- <s:complexType>
<!-- <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="GetLanguagesResult"
type="tns:LanguagesResult" />
</s:sequence>
</s:complexType>
</s:element>
<!-- <s:complexType name="LanguagesResult">
<!-- <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="Languages"
type="tns:ArrayOfLanguageEntry" />
<s:element minOccurs="1" maxOccurs="1" name="ErrorCode" type="s:int" />
<s:element minOccurs="0" maxOccurs="1" name="ErrorMessage" type="s:string" />
</s:sequence>
```

```
</s:complexType>
<!-- <s:complexType name="ArrayOfLanguageEntry">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="unbounded" name="LanguageEntry" nillable="true"
type="tns:LanguageEntry" />
  </s:sequence>
</s:complexType>
-->
<!-- <s:complexType name="LanguageEntry">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="1" name="LanguageKey" type="s:string" />
    <s:element minOccurs="0" maxOccurs="1" name="Name" type="s:string" />
  </s:sequence>
</s:complexType>
-->
</s:schema>
</wsdl:types>
-->
<wsdl:message name="LoginSoapIn">
  <wsdl:part name="parameters" element="tns:Login" />
</wsdl:message>
-->
<wsdl:message name="LoginSoapOut">
  <wsdl:part name="parameters" element="tns:LoginResponse" />
</wsdl:message>
-->
<wsdl:message name="ProcessTextSoapIn">
  <wsdl:part name="parameters" element="tns:ProcessText" />
</wsdl:message>
-->
<wsdl:message name="ProcessTextSoapOut">
  <wsdl:part name="parameters" element="tns:ProcessTextResponse" />
</wsdl:message>
-->
<wsdl:message name="ProcessTextForOfficeSoapIn">
  <wsdl:part name="parameters" element="tns:ProcessTextForOffice" />
</wsdl:message>
-->
<wsdl:message name="ProcessTextForOfficeSoapOut">
  <wsdl:part name="parameters" element="tns:ProcessTextForOfficeResponse" />
</wsdl:message>
-->
<wsdl:message name="GetUpdatedTextSoapIn">
  <wsdl:part name="parameters" element="tns:GetUpdatedText" />
</wsdl:message>
-->
<wsdl:message name="GetUpdatedTextSoapOut">
  <wsdl:part name="parameters" element="tns:GetUpdatedTextResponse" />
</wsdl:message>
-->
<wsdl:message name="GetConfigurationsSoapIn">
  <wsdl:part name="parameters" element="tns:GetConfigurations" />
</wsdl:message>
-->
<wsdl:message name="GetConfigurationsSoapOut">
  <wsdl:part name="parameters" element="tns:GetConfigurationsResponse" />
</wsdl:message>
-->
<wsdl:message name="ValidateAuthenticationSoapIn">
  <wsdl:part name="parameters" element="tns:ValidateAuthentication" />
</wsdl:message>
```

```
<wsdl:message name="ValidateAuthenticationSoapOut">
  <wsdl:part name="parameters" element="tns:ValidateAuthenticationResponse" />
</wsdl:message>
<wsdl:message name="GetLanguagesSoapIn">
  <wsdl:part name="parameters" element="tns:GetLanguages" />
</wsdl:message>
<wsdl:message name="GetLanguagesSoapOut">
  <wsdl:part name="parameters" element="tns:GetLanguagesResponse" />
</wsdl:message>
<wsdl:portType name="LiveServiceSoap">
  <wsdl:operation name="Login">
    <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Login user and get
authentication session key.</wsdl:documentation>
    <wsdl:input message="tns:LoginSoapIn" />
    <wsdl:output message="tns:LoginSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="ProcessText">
    <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Process text and get
result with link (live.lingulab.de) to processing details.</wsdl:documentation>
    <wsdl:input message="tns:ProcessTextSoapIn" />
    <wsdl:output message="tns:ProcessTextSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="ProcessTextForOffice">
    <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Process text for Office
AddIn and get result with link (live.lingulab.de) to processing details.</wsdl:documentation>
    <wsdl:input message="tns:ProcessTextForOfficeSoapIn" />
    <wsdl:output message="tns:ProcessTextForOfficeSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="GetUpdatedText">
    <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Get updated text and
results with link (live.lingulab.de) to processing details.</wsdl:documentation>
    <wsdl:input message="tns:GetUpdatedTextSoapIn" />
    <wsdl:output message="tns:GetUpdatedTextSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="GetConfigurations">
    <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Get available
processing configurations.</wsdl:documentation>
    <wsdl:input message="tns:GetConfigurationsSoapIn" />
    <wsdl:output message="tns:GetConfigurationsSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="ValidateAuthentication">
    <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Validate
authentication key.</wsdl:documentation>
    <wsdl:input message="tns:ValidateAuthenticationSoapIn" />
    <wsdl:output message="tns:ValidateAuthenticationSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="GetLanguages">
```

```
<wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Get available
processing languages.</wsdl:documentation>
<wsdl:input message="tns:GetLanguagesSoapIn" />
<wsdl:output message="tns:GetLanguagesSoapOut" />
</wsdl:operation>
</wsdl:portType>
<wsdl:binding name="LiveServiceSoap" type="tns:LiveServiceSoap">
<soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
<wsdl:operation name="Login">
<soap:operation soapAction="https://api.lingulab.net/LiveService.asmx/Login"
style="document" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="ProcessText">
<soap:operation soapAction="https://api.lingulab.net/LiveService.asmx/ProcessText"
style="document" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="ProcessTextForOffice">
<soap:operation soapAction="https://api.lingulab.net/LiveService.asmx/ProcessTextForOffice"
style="document" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetUpdatedText">
<soap:operation soapAction="https://api.lingulab.net/LiveService.asmx/GetUpdatedText"
style="document" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
```

```
<wsc:operation name="GetConfigurations">
  <soap:operation soapAction="https://api.lingulab.net/LiveService.asmx/GetConfigurations"
style="document" />
</wsc:operation>
<wsc:input>
  <soap:body use="literal" />
</wsc:input>
<wsc:output>
  <soap:body use="literal" />
</wsc:output>
</wsc:operation>
<wsc:operation name="ValidateAuthentication">
  <soap:operation
soapAction="https://api.lingulab.net/LiveService.asmx/ValidateAuthentication"
style="document" />
</wsc:operation>
<wsc:input>
  <soap:body use="literal" />
</wsc:input>
<wsc:output>
  <soap:body use="literal" />
</wsc:output>
</wsc:operation>
<wsc:operation name="GetLanguages">
  <soap:operation soapAction="https://api.lingulab.net/LiveService.asmx/GetLanguages"
style="document" />
</wsc:operation>
<wsc:input>
  <soap:body use="literal" />
</wsc:input>
<wsc:output>
  <soap:body use="literal" />
</wsc:output>
</wsc:operation>
</wsc:binding>
<wsc:binding name="LiveServiceSoap12" type="tns:LiveServiceSoap">
  <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
</wsc:binding>
<wsc:operation name="Login">
  <soap12:operation soapAction="https://api.lingulab.net/LiveService.asmx/Login"
style="document" />
</wsc:operation>
<wsc:input>
  <soap12:body use="literal" />
</wsc:input>
<wsc:output>
  <soap12:body use="literal" />
</wsc:output>
</wsc:operation>
<wsc:operation name="ProcessText">
  <soap12:operation soapAction="https://api.lingulab.net/LiveService.asmx/ProcessText"
style="document" />
</wsc:operation>
<wsc:input>
```

```
<soap12:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap12:body use="literal" />
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="ProcessTextForOffice">
<soap12:operation
soapAction="https://api.lingulab.net/LiveService.asmx/ProcessTextForOffice" style="document"
/>
<wsdl:input>
<soap12:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap12:body use="literal" />
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetUpdatedText">
<soap12:operation soapAction="https://api.lingulab.net/LiveService.asmx/GetUpdatedText"
style="document" />
<wsdl:input>
<soap12:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap12:body use="literal" />
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetConfigurations">
<soap12:operation soapAction="https://api.lingulab.net/LiveService.asmx/GetConfigurations"
style="document" />
<wsdl:input>
<soap12:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap12:body use="literal" />
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="ValidateAuthentication">
<soap12:operation
soapAction="https://api.lingulab.net/LiveService.asmx/ValidateAuthentication"
style="document" />
<wsdl:input>
<soap12:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap12:body use="literal" />
</wsdl:output>
</wsdl:operation>
```

```
<wsdl:operation name="GetLanguages">
  <soap12:operation soapAction="https://api.lingulab.net/LiveService.asmx/GetLanguages"
style="document" />
  <wsdl:input>
    <soap12:body use="literal" />
  </wsdl:input>
  <wsdl:output>
    <soap12:body use="literal" />
  </wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:service name="LiveService">
  <wsdl:port name="LiveServiceSoap" binding="tns:LiveServiceSoap">
    <soap:address location="http://api.lingulab.net/LiveService.asmx" />
  </wsdl:port>
  <wsdl:port name="LiveServiceSoap12" binding="tns:LiveServiceSoap12">
    <soap12:address location="http://api.lingulab.net/LiveService.asmx" />
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

2.3 Web Service Architecture

Web Service provides following main blocks:

- Authentication block: allows to authenticate user and provide to him credentials ([Login](#) web-method);
- Validation block: allows to check authentication key ([ValidateAuthentication](#) web-method);
- Processing Languages block: allows to get existing languages for processing ([GetLanguages](#) web-method)
- Configuration block: allows to get existing configurations of modules ([GetConfigurations](#) web-method);
- LinguLab Live Endpoint: provides access to user for LinguLab Live measure system ([ProcessText](#) web-method);
- Update edited text block: provides functionality of getting latest version of processed text ([GetUpdatedText](#) web-method).

Communication between client and LinguLab live web service:



Users should provide LinguLab account information to web service client (L1) and client should use these credentials for access to web service (L2).

Following described steps of using web service:

- Client calls [Login](#) web-method (L2) with account data and get back authentication key. The key is valid till web service authentication session expired. Next login the same account is not allowed when session is not expired. Client should store authentication key for current user and use it when it is not expired. Login web-method returns authentication key or error code (see Login web-method error code list).
- [ValidateAuthentication](#) web-method should be used for checking current authentication key. Method returns integer value which represents status of authentication key (see [ValidateAuthentication](#) web-method result code list).
- [GetLanguages](#) web-method returns processing language list. Processing languages can be get for future using in [GetConfigurations](#) and [ProcessText](#) method.
- Client can get processing configuration list for current user account by calling [GetConfigurations](#) web-method.
- [ProcessText](#) web-method should be used for text processing and getting text measure result.
- Having processed text with web service user can go to LinguLab for text editing (L3). Client can get latest text version by calling [GetUpdatedText](#) web-method.

3 Login web-method

3.1 Overview

Login web-method should be used for getting authentication session key. Authentication key is valid for 30 minutes and use sliding expiration. Having got authentication key it must be used in [GetConfigurations](#), [GetLanguages](#), [ProcessText](#) and [ValidateAuthentication](#) web-methods of Live web service. New authentication key cannot be get till previous is still valid. Therefore web service client should store authentication key for current username and use it till it will be expired.

Web-method returns authentication key in case of combination of username and password is valid and user has required role rights. In case of error occurred web-method will return error code and error message.

3.2 Error codes

Login web-method error code list:

- 0: Succeed;
- 101: Authentication failed;
- 102: Authentication session has already opened. You cannot open new session till old is not expired;
- 103: Too many login tries. Login tries limit was reached;
- 104: Wrong plugin key;
- 300: Internal error.

3.3 Request and response

The following is a sample SOAP 1.2 request and response. The placeholders shown need to be replaced with actual values.

```
POST /LiveService.asmx HTTP/1.1
Host: api.lingulab.net
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <Login xmlns="https://api.lingulab.net/LiveService.asmx">
      <userName>string</userName>
      <password>string</password>
      <pluginId>string</pluginId>
    </Login>
  </soap12:Body>
</soap12:Envelope>
```

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
```

```
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <LoginResponse
xmlns="https://api.lingulab.net/LiveService.asmx">
      <LoginResult>
        <AuthenticationKey>string</AuthenticationKey>
        <FullName>string</FullName>
        <RoleName>string</RoleName>
        <ErrorCode>int</ErrorCode>
        <ErrorMessage>string</ErrorMessage>
      </LoginResult>
    </LoginResponse>
  </soap12:Body>
</soap12:Envelope>
```

3.4 Using with .NET Framework 2.0

.NET Framework 2.0 client example (C#):

```
LiveService service = new LiveService();
LoginResult authResult = service.Login(txtUsername.Text, txtPassword.Text, null);

if (authResult.ErrorCode == 0)
{
    //Authentication Key can be used in Web Service methods: authResult.AuthenticationKey
}
else
{
    //Login failed
}
```

4 ValidateAuthentication web-method

4.1 Overview

`ValidateAuthentication` web-method should be used for checking authentication key. Web-method returns "0" if authentication key is valid.

4.2 Error codes

`ValidateAuthentication` web method result code list:

- 0: Authentication key is valid;
- 111: Authentication key is not valid;
- 112: Authentication key was expired;
- 210: Wrong parameter. Empty authentication key;
- 310: Internal error.

4.3 Request and response

The following is a sample SOAP 1.2 request and response. The placeholders shown need to be replaced with actual values.

```
POST /LiveService.asmx HTTP/1.1
Host: api.lingulab.net
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <ValidateAuthentication
xmlns="https://api.lingulab.net/LiveService.asmx">
      <authenticationKey>string</authenticationKey>
    </ValidateAuthentication>
  </soap12:Body>
</soap12:Envelope>
```

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <ValidateAuthenticationResponse
xmlns="https://api.lingulab.net/LiveService.asmx">
      <ValidateAuthenticationResult>int</ValidateAuthenticationResult>
    </ValidateAuthenticationResponse>
  </soap12:Body>
</soap12:Envelope>
```

Using with .NET Framework 2.0

.NET Framework 2.0 client example (C#):

```
LiveService service = new LiveService();  
int validationResult = service.ValidateAuthentication(authKey);  
if (validationResult != 0)  
{  
    //Authentication key is not valid. Process error code for details  
}  
else  
{  
    //Authentication key is valid  
}
```

5 GetLanguages web-method

5.1 Overview

[GetLanguages](#) web-method should be used for getting processing language list. Web-method returns “0” if authentication key is valid.

5.2 Error codes

[GetLanguages](#) web method result code list:

- 0: Succeed;
- 121: Authentication failed;
- 122: Authentication key was expired;
- 123: Lack of permission;
- 220: Wrong parameter. Some fields in input data are missed, detailed information should be provided in error message;
- 320: Internal error.

5.3 Request and response

The following is a sample SOAP 1.2 request and response. The placeholders shown need to be replaced with actual values.

```
POST /LiveService.asmx HTTP/1.1
Host: api.lingulab.net
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetLanguages
xmlns="https://api.lingulab.net/LiveService.asmx">
      <authenticationKey>string</authenticationKey>
    </GetLanguages>
  </soap12:Body>
</soap12:Envelope>
```

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetLanguagesResponse
xmlns="https://api.lingulab.net/LiveService.asmx">
      <GetLanguagesResult>
        <Languages>
          <LanguageEntry>
            <LanguageKey>string</LanguageKey>
            <Name>string</Name>
```

```
</LanguageEntry>
<LanguageEntry>
  <LanguageKey>string</LanguageKey>
  <Name>string</Name>
</LanguageEntry>
</Languages>
<ErrorCode>int</ErrorCode>
<ErrorMessage>string</ErrorMessage>
</GetLanguagesResult>
</GetLanguagesResponse>
</soap12:Body>
</soap12:Envelope>
```

Using with .NET Framework 2.0

.NET Framework 2.0 client example (C#):

```
LiveService service = new LiveService();
LanguagesResult langs = service.GetLanguages(authenticationKey);
if (langs.ErrorCode != 0)
{
  //Authentication key is not valid. Process error code for details
}
else
{
  //Authentication key is valid
}
```

6 GetConfigurations web-method

6.1 Overview

[GetConfigurations](#) web-method should be used for getting configuration list. Configuration list can be used for choosing required configuration of modules for processing. [ProcessText](#) web-method requires configuration name which can be get from configuration list.

Web-method returns list of configuration entries with ids and names of configuration. Also entries contain information about keywords supporting by configuration. In case of error occurred [ErrorCode](#) and [ErrorMessage](#) contain information about error.

6.2 Error codes

[GetConfigurations](#) web-method error code list:

- 0: Succeed;
- 121: Authentication failed;
- 122: Authentication key was expired;
- 123: Lack of permission;
- 220: Wrong parameter. Some fields in input data are missed, detailed information should be provided in error message;
- 223: Wrong language key;
- 320: Internal error.

6.3 Processing configurations

Web-Text Grundform category (default):

- Quantität (Satzlänge and Wortlänge modules)
- Stil (Satzkonstruktionen, Modalität, Passivkonstruktionen, Zeit and Nominalstil modules)
- Wortschatz (Worthäufigkeit, Hilfsverben, Füllwörter, Abkürzung and Adjektivhäufung modules)
- Scanbarkeit (Teaser, Absätze, Überschriften, Listen, Strukturierung and Hervorhebungen modules)
- Interaktivität (Direkte and Sprache Kontakt modules)
- Prägnanz (Textlänge, Suchmaschinenoptimierung and Info-Vermittlung modules)

Expertentext Web category:

- Quantität (Satzlänge module)
- Stil (Satzkonstruktionen, Modalität, Passivkonstruktionen, Zeit and Nominalstil modules)
- Wortschatz (Füllwörter, Hilfsverben, Adjektivhäufung and Abkürzung modules)
- Scanbarkeit (Teaser, Absätze, Überschriften, Listen, Strukturierung and Hervorhebungen modules)
- Prägnanz (Textlänge and Suchmaschinenoptimierung modules)

Expertentext Print category:

- Quantität (Satzlänge module)

- Stil (Satzkonstruktionen, Modalität, Passivkonstruktionen, Zeit and Nominalstil module)
- Wortschatz (Worthäufigkeit, Überschriften, Füllwörter, Strukturierung, Adjektivhäufung, Absätze, Hilfsverben, Listen, Abkürzung and Info-Vermittlung modules)

Verkaufstext Web category:

- Quantität (Satzlänge and Wortlänge modules)
- Stil (Satzkonstruktionen, Nominalstil, Passivkonstruktionen and Modalität modules)
- Wortschatz (Worthäufigkeit, Hilfsverben, Füllwörter, Abkürzung and Adjektivhäufung modules)
- Scanbarkeit (Überschriften, Listen, Strukturierung, Hervorhebungen and Absätze modules)
- Interaktivität (Direkte Sprache and Kontakt modules)
- Prägnanz (Textlänge and Suchmaschinenoptimierung modules)

Verkaufstext Print category:

- Quantität (Satzlänge and Wortlänge modules)
- Stil (Satzkonstruktionen, Nominalstil, Passivkonstruktionen and Modalität modules)
- Wortschatz (Worthäufigkeit, Überschriften, Füllwörter, Strukturierung, Adjektivhäufung, Absätze, Hilfsverben, Listen, Abkürzung and Hervorhebungen modules)
- Interaktivität (Direkte Sprache and Kontakt modules)
- Prägnanz (Textlänge module)

Produktbeschreibung Web category:

- Quantität (Satzlänge and Wortlänge modules)
- Stil (Satzkonstruktionen, Modalität, Passivkonstruktionen, Zeit and Nominalstil modules)
- Wortschatz (Worthäufigkeit, Hilfsverben, Füllwörter and Abkürzung modules)
- Scanbarkeit (Überschriften, Listen, Strukturierung, Hervorhebungen and Absätze modules)
- Interaktivität (Direkte Sprache module)
- Prägnanz (Textlänge and Suchmaschinenoptimierung modules)

Web-Text intern category:

- Quantität (Satzlänge and Wortlänge modules)
- Stil (Satzkonstruktionen, Zeit, Passivkonstruktionen, Füllwörter, Nominalstil, Adjektivhäufung, Modalität and Hilfsverben modules)
- Scanbarkeit (Teaser, Absätze, Überschriften, Listen, Strukturierung and Hervorhebungen modules)
- Interaktivität (Direkte Sprache module)
- Prägnanz (Textlänge module)

Pressemitteilung Web category:

- Quantität (Satzlänge and Wortlänge module)
- Stil (Satzkonstruktionen, Modalität, Passivkonstruktionen, Zeit and Nominalstil modules)
- Wortschatz (Worthäufigkeit, Hilfsverben, Füllwörter, Abkürzung and Adjektivhäufung modules)
- Scanbarkeit (Teaser, Absätze, Überschriften, Listen, Strukturierung and Hervorhebungen modules)
- Interaktivität (Direkte Sprache and Kontakt modules)
- Prägnanz (Textlänge and Suchmaschinenoptimierung modules)

Verwaltungstext Web category:

- Quantität (Satzlänge and Wortlänge modules)
- Stil (Satzkonstruktionen, Modalität, Passivkonstruktionen, Zeit and Nominalstil modules)
- Wortschatz (Worthäufigkeit, Hilfsverben, Füllwörter, Abkürzung and Adjektivhäufung modules)
- Scanbarkeit (Teaser, Absätze, Überschriften, Listen, Strukturierung and Hervorhebungen modules)
- Interaktivität (Direkte Sprache module)
- Prägnanz (Textlänge and Suchmaschinenoptimierung modules)

6.4 Request and response

The following is a sample SOAP 1.2 request and response. The placeholders shown need to be replaced with actual values.

```
POST /LiveService.asmx HTTP/1.1
Host: api.lingulab.net
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetConfigurations
xmlns="https://api.lingulab.net/LiveService.asmx">
      <authenticationKey>string</authenticationKey>
      <languageKey>string</languageKey>
    </GetConfigurations>
  </soap12:Body>
</soap12:Envelope>
```

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
```

```
<GetConfigurationsResponse
xmlns="https://api.lingulab.net/LiveService.asmx">
  <GetConfigurationsResult>
    <Configurations>
      <ConfigurationEntry>
        <Id>string</Id>
        <Name>string</Name>
        <IsKeywordsSupported>boolean</IsKeywordsSupported>
      </ConfigurationEntry>
      <ConfigurationEntry>
        <Id>string</Id>
        <Name>string</Name>
        <IsKeywordsSupported>boolean</IsKeywordsSupported>
      </ConfigurationEntry>
    </Configurations>
    <ErrorCode>int</ErrorCode>
    <ErrorMessage>string</ErrorMessage>
  </GetConfigurationsResult>
</GetConfigurationsResponse>
</soap12:Body>
</soap12:Envelope>
```

6.5 Using with .NET Framework 2.0

.NET Framework 2.0 client example (C#):

```
LiveService service = new LiveService();
string authenticationKey= service.Login("user", "password");

ConfigurationResult confResult = service.GetConfigurations(authenticationKey, languageKey);
if (confResult.ErrorCode != 0)
{
    //Error occurred. Display error.
    return;
}
ConfigurationEntry[] categories = confResult.Configurations;

if (categories == null || categories.Length == 0)
{
    //error of getting category list
    throw new ApplicationException("Empty Category List.");
}

foreach (ConfigurationEntry entry in categories)
{
    if (entry.IsKeywordsSupported)
    {
        //Allow to add keywords
    }
    //Set categoryKey field
}
```

7 Processtext web-method

7.1 Overview

ProcessText web-method should be used for processing text with LinguLab Live. Web-method require HTML text for processing, configuration identifier and authentication key. Search keywords are optional. Language key should be set but if it does not set default language will be used for processing. Web-method returns **ProcessTextResult** object with includes result identifier, text measure (integer value from 0 to 10), HTML of measure image with details and link to processing details page. Text can be optimized on LinguLab live web site. In case of error occurred **ErrorCode** and **ErrorMessage** contain information about error. “lingulabCustom” attribute can be used for storing custom data of tag for future parsing on client side. Current attribute will not be removed from html when text edited (See example in “GetUpdatedText web-method” chapter).

HTML of measure image with details can look like:



7.2 Processing text format

Processing text tags will be transformed by default in following way (all html tags will be scraped from processing text except this list):

- h1 - headline
- h2 - subheadline
- h3 - teaser
- p - text
- ul - bullet list
- ol - number list

Other html tag will be processed as Unknown.

Web service client can provide custom section type settings with **SectionTypeSetting** property of **ProcessTextData** parameter. The property is optional. In case this property is not set default settings will be used. Example of custom settings (C#):

```
inputData.SectionTypeSettings = new SectionTypeSetting();
inputData.SectionTypeSettings.Headline = "h2";
inputData.SectionTypeSettings.SubHeadline = "h3";
inputData.SectionTypeSettings.Teaser = "h4";
inputData.LanguageKey = langKey;
inputData.SectionTypeSettings.Text = "p";
inputData.SectionTypeSettings.BulletList = new ListSetting();
inputData.SectionTypeSettings.BulletList.ItemTag = "li";
inputData.SectionTypeSettings.BulletList.ListTag = "ul";
inputData.SectionTypeSettings.NumberList = new ListSetting();
inputData.SectionTypeSettings.NumberList.ItemTag = "li";
inputData.SectionTypeSettings.NumberList.ListTag = "ol";
```

The char length of processing text cannot be more than allowed for current user account role.

7.3 Error codes

ProcessText web-method error code list:

- 0: Succeed;
- 131: Authentication failed;
- 132: Authentication key was expired;
- 133: Lack of permission;
- 230: Wrong parameter. Some fields in input data are missed, detailed information should be provided in error message;
- 231: Processing text length error;
- 232: Getting configuration file failed;
- 233: Wrong language;
- 330: Internal error;
- 331: Storing data failed.

7.4 Request and response

The following is a sample SOAP 1.2 request and response. The placeholders shown need to be replaced with actual values.

```
POST /LiveService.asmx HTTP/1.1
Host: api.lingulab.net
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <ProcessText
xmlns="https://api.lingulab.net/LiveService.asmx">
      <inputData>
        <Text>string</Text>
        <ConfigurationId>string</ConfigurationId>
        <LanguageKey>string</LanguageKey>
        <SearchKeyword1>string</SearchKeyword1>
        <SearchKeyword2>string</SearchKeyword2>
        <SearchKeyword3>string</SearchKeyword3>
        <SectionTypeSettings>
```

```
<Headline>string</Headline>
<SubHeadline>string</SubHeadline>

<Teaser>string</Teaser>

<Text>string</Text>
<BulletList>
  <ListTag>string</ListTag>
  <ItemTag>string</ItemTag>
</BulletList>
<NumberList>
  <ListTag>string</ListTag>
  <ItemTag>string</ItemTag>
</NumberList>
</SectionTypeSettings>
</inputData>
<authenticationKey>string</authenticationKey>
</ProcessText>
</soap12:Body>
</soap12:Envelope>
```

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <ProcessTextResponse
xmlns="https://api.lingulab.net/LiveService.asmx">
      <ProcessTextResult>
        <ResultId>int</ResultId>
        <Measure>int</Measure>
        <MeasureStarsHtml>string</MeasureStarsHtml>
        <LinkOnResultPage>string</LinkOnResultPage>
        <ErrorCode>int</ErrorCode>
        <ErrorMessage>string</ErrorMessage>
      </ProcessTextResult>
    </ProcessTextResponse>
  </soap12:Body>
</soap12:Envelope>
```

7.5 Using with .NET Framework 2.0

.NET Framework 2.0 client example (C#):

```
ProcessTextData inputData = new ProcessTextData();
inputData.Text = "Some text";
inputData.ConfigurationName = categoryKey;
inputData.LanguageKey = langKey;
inputData.SearchKey1 = "some Key";

ProcessTextResult result = service.ProcessText(inputData, authenticationKey);
if (result.ErrorCode == 0)
{
  //do something on no error
}
else
{
  //proces ErrorCode
}
```

```
}
```

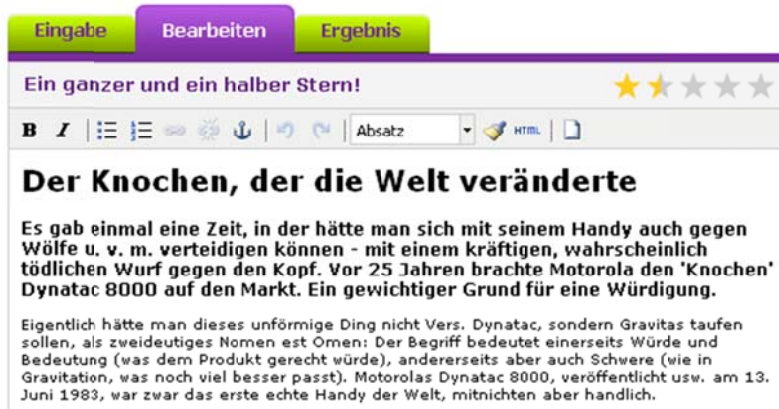
8 GetUpdatedText web-method

8.1 Overview

`GetUpdatedText` web-method should be used for getting back edited text from LinguLab live web site. Web-method requires result key and valid authentication key.

Web-method returns `UpdatedTextResult` object with includes result identifier, latest version of raw text, version number, text measure (integer value from 0 to 10), HTML of measure image with details and link to processing details page. In case of error occurred `ErrorCode` and `ErrorMessage` contain information about error.

User can go to LinguLab live by link to processing details returned by `ProcessText` web-method and edit text on “Bearbeiten” (available only for web service link) tab:



Having edited text user can click “Zeige Ergebnis” button at the bottom of the page:



Text was updated and stored. Optimize text till it has required measure result. Then `GetUpdatedText` web-method can be used for getting the latest version of text. Use “lingulabCustom” attribute for storing custom data (for data relation on client side). For example:

```
<h1 lingulabCustom="1b5b669a-dabe-4986-96d4-7ac8e458a5f3_0">
  Headline text
</h1>
<h3 lingulabCustom="1b5b669a-dabe-4986-96d4-7ac8e458a5f3_1">
  Teaser text.<br /><br />
  Some text.
</h3>
<div lingulabCustom="1b5b669a-dabe-4986-96d4-7ac8e458a5f3_3">
  Simple text
</div>
```

The “lingulabCustom” attribute contains id of text section and index of element separated by “_” in current example. Value of this attribute will be exist when text come back with [GetUpdatedText](#) web-method and can be used by client application.

Note: Having clicked “Eingabe” link new text processing will be started and web service text will not be updated.

8.2 Error codes

GetUpdatedText web-method error code list:

- 0: Succeed;
- 141: Authentication failed;
- 142: Authentication key was expired;
- 143: Lack of permission;
- 240: Wrong parameter. Empty result key or authentication key;
- 241: Wrong result key. Could not get results by provided result key;
- 340: Internal error;

8.3 Request and response

The following is a sample SOAP 1.2 request and response. The placeholders shown need to be replaced with actual values.

```
POST /LiveService.asmx HTTP/1.1
Host: api.lingulab.net
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetUpdatedText
xmlns="https://api.lingulab.net/LiveService.asmx">
      <resultKey>int</resultKey>
      <authenticationKey>string</authenticationKey>
    </GetUpdatedText>
  </soap12:Body>
</soap12:Envelope>
```

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetUpdatedTextResponse
xmlns="https://api.lingulab.net/LiveService.asmx">
      <GetUpdatedTextResult>
        <ResultId>int</ResultId>
        <Measure>int</Measure>
      </GetUpdatedTextResult>
    </GetUpdatedTextResponse>
  </soap12:Body>
</soap12:Envelope>
```

```
<MeasureStarsHtml>string</MeasureStarsHtml>  
<LinkOnResultPage>string</LinkOnResultPage>  
<ErrorCode>int</ErrorCode>  
<ErrorMessage>string</ErrorMessage>  
<Version>int</Version>  
<RawText>string</RawText>  
</GetUpdatedTextResult>  
</GetUpdatedTextResponse>  
</soap12:Body>  
</soap12:Envelope>
```

8.4 Using with .NET Framework 2.0

.NET Framework 2.0 client example (C#):

```
UpdatedTextResult updateResult = service.GetUpdatedText(resultKey, authenticationKey);  
  
if (updateResult.ErrorCode == 0)  
{  
    //Display results  
}  
else  
{  
    //proces ErrorCode  
}
```

9 Example Implementations

9.1 ASP.NET 2.0

Example of using Live Web Service in ASP.NET website.

Live Web Service web reference should be added to website project:



Simple page contains login text boxes, drop-down with text processing categories, text area for text, keyword field, "Start Analyzing" button and image for displaying results.

First enter your login data:

Live Web Service Sample

LinguLab Authentication data.
Username:
Password:
[Login](#)

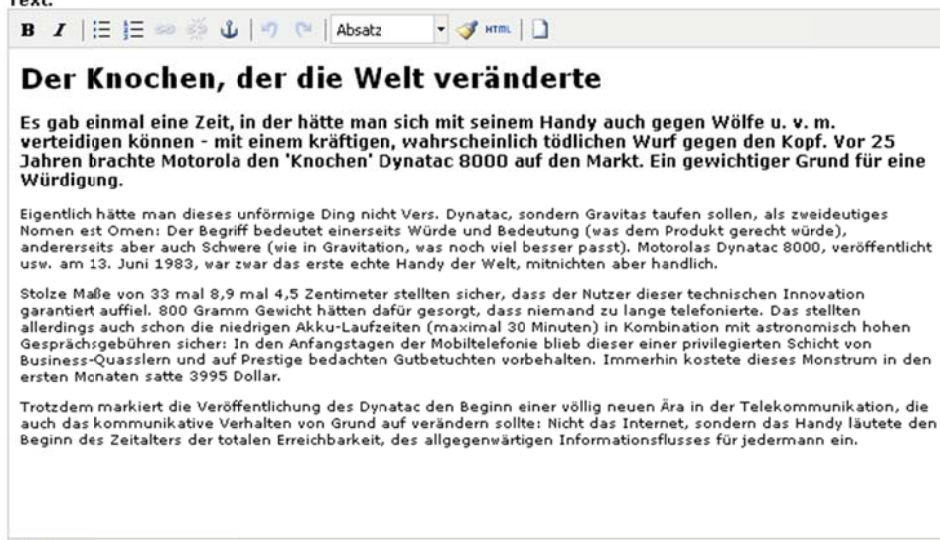
Then select category, enter keywords and text for processing. Click “Start Analyzing” button. “Update” button should be displayed. It can be used for updating text from LinguLab live. Result details can be displayed by clicking on image with stars:

Suchmaschinenrelevanz

Geben Sie maximal drei Keywords ein. Die Suchmaschinenrelevanz des eingegebenen Textes wird auf Grundlage dieser Keywords ausgewertet.

1. 2. 3.

Text:



Der Knochen, der die Welt veränderte

Es gab einmal eine Zeit, in der hätte man sich mit seinem Handy auch gegen Wölfe u. v. m. verteidigen können - mit einem kräftigen, wahrscheinlich tödlichen Wurf gegen den Kopf. Vor 25 Jahren brachte Motorola den 'Knochen' Dynatac 8000 auf den Markt. Ein gewichtiger Grund für eine Würdigung.

Eigentlich hätte man dieses unförmige Ding nicht Vers. Dynatac, sondern Gravitac taufen sollen, als zweideutiges Nomen est Omen: Der Begriff bedeutet einerseits Würde und Bedeutung (was dem Produkt gerecht würde), andererseits aber auch Schwere (wie in Gravitation, was noch viel besser passt). Motorolas Dynatac 8000, veröffentlicht usw. am 13. Juni 1983, war zwar das erste echte Handy der Welt, mitnichten aber handlich.

Stolze Maße von 33 mal 8,9 mal 4,5 Zentimeter stellten sicher, dass der Nutzer dieser technischen Innovation garantiert auffiel. 800 Gramm Gewicht hätten dafür gesorgt, dass niemand zu lange telefonierte. Das stellten allerdings auch schon die niedrigen Akku-Laufzeiten (maximal 30 Minuten) in Kombination mit astronomisch hohen Gesprächsgebühren sicher: In den Anfangstagen der Mobiltelefonie blieb dieser einer privilegierten Schicht von Business-Quasslern und auf Prestige bedachten Gutbetuchten vorbehalten. Immerhin kostete dieses Monstrum in den ersten Monaten satte 3995 Dollar.

Trotzdem markiert die Veröffentlichung des Dynatac den Beginn einer völlig neuen Ära in der Telekommunikation, die auch das kommunikative Verhalten von Grund auf verändern sollte: Nicht das Internet, sondern das Handy läutete den Beginn des Zeitalters der totalen Erreichbarkeit, des allgegenwärtigen Informationsflusses für jedermann ein.

Pfad:
[Update](#) [Start Analyzing](#)

Result:



Code of page following:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs"
Inherits="_Default"
ValidateRequest="false" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>Live Web Service Sample</title>

  <script src="scripts/jquery.js" type="text/javascript"></script>

  <script src="scripts/slider.js" type="text/javascript"></script>

  <link href="css/main.css" rel="stylesheet" type="text/css" />
</head>
```

```
<body>
  <div id="content">
    <asp:Image ID="Image1" ImageUrl="~/images/header_bg.jpg" runat="server" />
    <form id="form1" runat="server">
      <h1 class="headline">
        Live Web Service Sample</h1>
      <p class="bodytext">
        LinguLab Authentication data.
        <br />
        Username:
        <asp:TextBox ID="txtUsername" CssClass="txt" runat="server"></asp:TextBox><br />
        Password:
        <asp:TextBox ID="txtPassword" TextMode="Password" CssClass="txt"
runat="server"></asp:TextBox><br />
        <asp:LinkButton ID="lnkLogin" runat="server"
OnClick="lnkLogin_Click">Login</asp:LinkButton>
      </p>
      <p class="bodytext">
        Bitte wählen Sie die Textgattung mit Hilfe des Dropdown-Menüs aus.
        <br />
        <br />
        <asp:DropDownList ID="drCategories" runat="server">
        </asp:DropDownList>
      </p>
      <p class="trenner" />
      <p class="bodytext" style="border: 1px solid rgb(98, 33, 129); margin: 0pt 20px 25px 0;
padding: 20px;">
        <b>Suchmaschinenrelevanz</b>
        <br />
        <br />
        Geben Sie maximal drei Keywords ein. Die Suchmaschinenrelevanz des eingegebenen
        Textes wird auf Grundlage dieser Keywords ausgewertet. <br />
        <br />
        1.<asp:TextBox ID="txtSEOKeyword1" CssClass="txt" runat="server"></asp:TextBox>
        2.<asp:TextBox ID="txtSEOKeyword2" CssClass="txt" runat="server"></asp:TextBox>
        3.<asp:TextBox ID="txtSEOKeyword3" CssClass="txt" runat="server"></asp:TextBox>
      </p>
      <p class="trenner" />
      <div>
        <asp:Label ID="lblVesion" runat="server" Text=""></asp:Label><br />

        <script language="javascript" type="text/javascript"
src="scripts/tiny_mce/tiny_mce.js"></script>

        <script language="javascript" type="text/javascript">
          tinyMCE.init({
            mode: "textareas",
            language: "de",
            theme: "advanced",
            theme_advanced_buttons1:
"bold,italic,|,bullist,numlist,link,unlink,anchor,|,undo,redo,|,formatselect,cleanup,code,|,newdoc
ument",
            theme_advanced_buttons2: "",
            theme_advanced_buttons3: "",
            theme_advanced_toolbar_location: "top",
            theme_advanced_toolbar_align: "left",
            theme_advanced_blockformats: "p,h2,h3,h4",
```

```
theme_advanced_statusbar_location: "bottom",
valid_elements: ""
    + "a[href],"
    + "h1,"
    + "h2,"
    + "h3,h4,h5,h6,"
    + "div,span,font,label,small,u,i"
    + "img[src],"
    + "li,"
    + "ol,"
    + "p,"
    + "strong/b,"
    + "ul"
});
</script>
<b>Text:</b>
<textarea id="txtText" runat="server" style="width: 687px; height: 400px" rows="10"
cols="5"></textarea>
</div>
<asp:LinkButton ID="InkUpdateText" runat="server" OnClick="InkUpdateText_Click"
Visible="false">Update</asp:LinkButton>
<asp:LinkButton ID="InkAnalyze" runat="server" OnClick="InkAnalyze_Click"
Visible="false">Start Analyzing</asp:LinkButton>
<asp:Label ID="lblErrorMessage" runat="server" Text="" Visible="true"
ForeColor="Red"></asp:Label>
<asp:Panel ID="pnlResultPanel" Visible="false" runat="server">
    Result: <span id="spMeasureStars" runat="server" />
</asp:Panel>
</form>
</div>
<br />
<br />
</body>
</html>
```

Code file:

```
using System;
using System.Collections.Generic;
using System.Web.UI.WebControls;
using live.lingulab.de;
using System.Configuration;

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            ToggleFunctionality(false);
        }
    }

    /// <summary>
    /// Authentication keys for users.

```

```
/// </summary>
private Dictionary<string, string> Authentications
{
    get
    {
        Dictionary<string, string> auth = Application["LiveAuthentications"] as Dictionary<string,
string>;
        return auth;
    }
    set
    {
        Application["LiveAuthentications"] = value;
    }
}

/// <summary>
/// Current result key.
/// </summary>
private int? ResultKey
{
    get
    {
        int? resultKey = (int?)ViewState["ResultKey"];
        return resultKey;
    }
    set
    {
        ViewState["ResultKey"] = value;
    }
}

private void ToggleFunctionality(bool enable)
{
    txtSEOKeyword1.Enabled = enable;
    txtSEOKeyword2.Enabled = enable;
    txtSEOKeyword3.Enabled = enable;
    drCategories.Enabled = enable;
    txtUsername.Enabled = !enable;
    txtPassword.Enabled = !enable;
    lnkLogin.Visible = !enable;
}

/// <summary>
/// Bind categories on page.
/// </summary>
private void BindCategories()
{
    //Create live web service instance
    LiveService service = new LiveService();

    //Get authentication key
    string authenticationKey = GetAuthenticationKey(service);
    if (authenticationKey.Length < 4) return;

    //Get configuration list
    ConfigurationResult confResult = service.GetConfigurations(authenticationKey);
    if (confResult.ErrorCode != 0)
```

```
{
    //Error occured
    pnlResultPanel.Visible = false;
    lblErrorMessage.Text = "Error of getting categories. Error Code:" + confResult.ErrorCode + "
Message: " + confResult.ErrorMessage;
    return;
}
ConfigurationEntry[] categories = confResult.Configurations;

if (categories == null || categories.Length == 0)
{
    //error of getting category list
    throw new ApplicationException("Empty Category List.");
}

drCategories.Items.Clear();
foreach (ConfigurationEntry entry in categories)
{
    if (entry.IsKeywordsSupported)
    {
        //Allow to add keywords
    }
    //Set categoryKey field
    drCategories.Items.Add(new ListItem(entry.Name, entry.Id));
}
drCategories.SelectedIndex = 0;
InkAnalyze.Visible = true;
}

/// <summary>
/// Returns authentication key if authentication succeed.
/// </summary>
/// <param name="service">LinguLab live Web service object.</param>
/// <returns>Authentication key. Empty string if not authenticated.</returns>
private string GetAuthenticationKey(LiveService service)
{
    string authKey = string.Empty;

    //get key from cach
    Dictionary<string, string> authentications = Authentications;
    if (authentications != null
        && authentications.ContainsKey(txtUsername.Text))
        authKey = authentications[txtUsername.Text];

    int validationResult = service.ValidateAuthentication(authKey);
    if (string.IsNullOrEmpty(authKey) || validationResult != 0)
    {
        //request new authentication key
        LoginResult authResult = service.Login(txtUsername.Text, txtPassword.Text);
        authKey = authResult.AuthenticationKey;
        if (authentications == null)
        {
            authentications = new Dictionary<string, string>();
        }

        authentications[txtUsername.Text] = authResult.AuthenticationKey;
    }
}
```

```
Authentications = authentications;

if (authResult.ErrorCode != 0)
{
    //error of getting authentication key
    pnlResultPanel.Visible = false;
    lblErrorMessage.Text = "Live Service Authentication Failed. Message: " +
Constants.LoginErrors[authResult.ErrorCode];
    txtUsername.Enabled = true;
    txtPassword.Enabled = true;
    lnkLogin.Visible = true;
}
else
{
    lblErrorMessage.Text = string.Empty;
}

}

return authKey;
}

/// <summary>
/// "Start Analyzing" link click.
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
protected void lnkAnalyze_Click(object sender, EventArgs e)
{
    //Create live web service instance
    LiveService service = new LiveService();
    lblVesion.Visible = false;
    //Get authentication key
    string authenticationKey = GetAuthenticationKey(service);
    if (authenticationKey.Length < 4) return;

    //Get selected category key
    string categoryKey = drCategories.SelectedValue;

    //Prepare processing data
    ProcessTextData inputData = new ProcessTextData();
    inputData.Text = txtText.Value;
    inputData.ConfigurationId = categoryKey;
    inputData.SearchKeyword1 = txtSEOKeyword1.Text;
    inputData.SearchKeyword2 = txtSEOKeyword2.Text;
    inputData.SearchKeyword3 = txtSEOKeyword3.Text;

    inputData.SectionTypeSettings = new SectionTypeSetting();
    inputData.SectionTypeSettings.Headline = "h2";
    inputData.SectionTypeSettings.SubHeadline = "h3";
    inputData.SectionTypeSettings.Teaser = "h4";
    inputData.SectionTypeSettings.Text = "p";
    inputData.SectionTypeSettings.BulletList = new ListSetting();
    inputData.SectionTypeSettings.BulletList.ItemTag = "li";
    inputData.SectionTypeSettings.BulletList.ListTag = "ul";
}
```

```
inputData.SectionTypeSettings.NumberList = new ListSetting();
inputData.SectionTypeSettings.NumberList.ItemTag = "li";
inputData.SectionTypeSettings.NumberList.ListTag = "ol";

//Process text
ProcessTextResult result = service.ProcessText(inputData, authenticationKey);

if (result.ErrorCode == 0)
{

    //Display results
    lblErrorMessage.Text = string.Empty;
    pnlResultPanel.Visible = true;

    spMeasureStars.InnerHtml = result.MeasureStarsHtml;
    ResultKey = result.ResultId;
    InkUpdateText.Visible = true;

}
else
{
    //proces ErrorCode
    pnlResultPanel.Visible = false;
    string message = "Unknown.";
    if (Constants.ProcessingErrors.ContainsKey(result.ErrorCode))
    {
        message = Constants.ProcessingErrors[result.ErrorCode];
    }
    lblErrorMessage.Text = message + " " + result.ErrorMessage;
    ResultKey = null;
    InkUpdateText.Visible = false;
}
}

/// <summary>
/// "Update" text link click.
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
protected void InkUpdateText_Click(object sender, EventArgs e)
{
    //Create live web service instance
    LiveService service = new LiveService();

    //Get authentication key
    string authenticationKey = GetAuthenticationKey(service);
    if (authenticationKey.Length < 4) return;

    int? resultKey = ResultKey;
    if (resultKey == null)
    {
        throw new ApplicationException("Result Key is null in view state");
    }
}
```

```
//Get updated text
UpdatedTextResult updateResult = service.GetUpdatedText(resultKey.Value,
authenticationKey);

if (updateResult.ErrorCode == 0)
{
    //Display results
    lblErrorMessage.Text = string.Empty;
    pnlResultPanel.Visible = true;
    txtText.Value = updateResult.RawText;
    lblVersion.Text = "Version: " + updateResult.Version;
    lblVersion.Visible = true;
    spMeasureStars.InnerHtml = updateResult.MeasureStarsHtml;
}
else
{
    //proces ErrorCode
    pnlResultPanel.Visible = false;
    lblErrorMessage.Text = updateResult.ErrorMessage;
}
}

/// <summary>
/// "Login" link click.
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
protected void lnkLogin_Click(object sender, EventArgs e)
{
    // Bind categories
    BindCategories();

    if (lblErrorMessage.Text.Length == 0)
    {
        ToggleFunctionality(true);
    }
}
}
```

Application error constants:

```
public class Constants
{
    /// <summary>
    /// Login error list.
    /// </summary>
    public readonly static Dictionary<int, string> LoginErrors = new Dictionary<int, string>();

    /// <summary>
    /// Processing error list.
    /// </summary>
    public readonly static Dictionary<int, string> ProcessingErrors = new Dictionary<int, string>();
    static Constants()
    {
        LoginErrors.Add(101, "Authentication failed");
    }
}
```

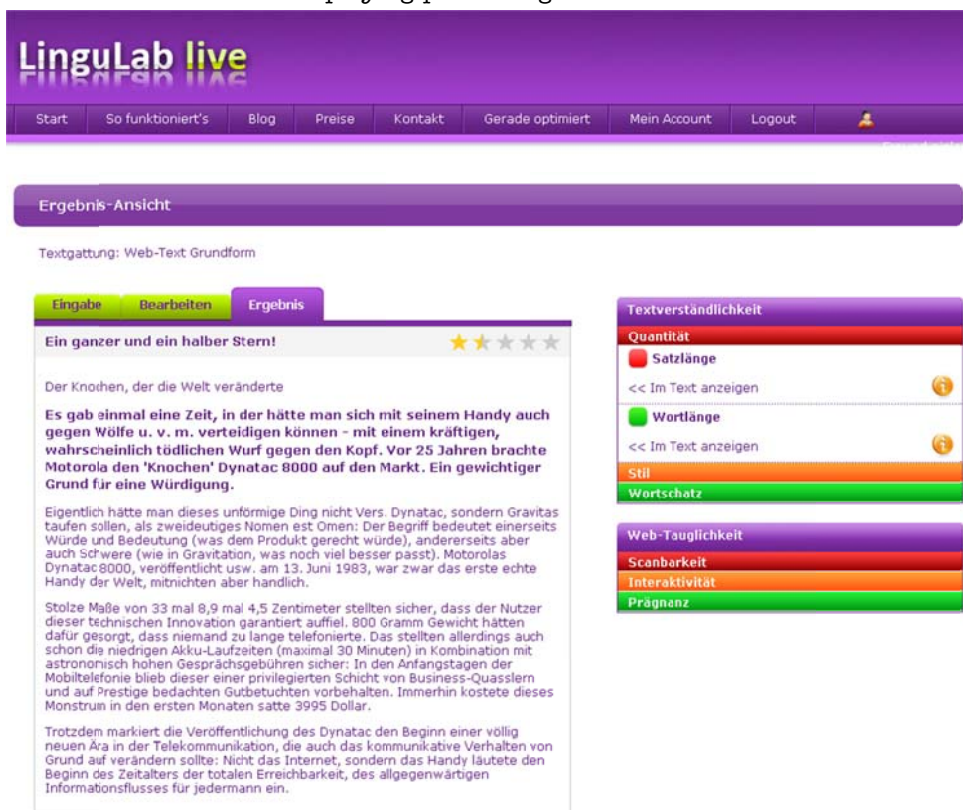
```

LoginErrors.Add(102, "Authentication session has already opened. You cannot open new
session till old is not expired.");
LoginErrors.Add(103, "Too many login tries. Login tries limit was reached.");
LoginErrors.Add(300, "Internal error");

ProcessingErrors.Add(131, "Authentication failed.");
ProcessingErrors.Add(132, "Authentication key was expired.");
ProcessingErrors.Add(133, "Lack of permission error.");
ProcessingErrors.Add(230, "Wrong parameter.");
ProcessingErrors.Add(231, "Processing text length error.");
ProcessingErrors.Add(232, "Getting configuration file failed.");
ProcessingErrors.Add(330, "Internal error.");
ProcessingErrors.Add(331, "Storing data failed.");
}
}

```

Go to returned link for displaying processing result details:



The screenshot shows the 'LinguLab live' web interface. At the top, there is a navigation menu with links: Start, So funktioniert's, Blog, Preise, Kontakt, Gerade optimiert, Mein Account, Logout, and a user icon. Below the menu is a purple header with the text 'Ergebnis-Ansicht'. Underneath, it indicates 'Textgattung: Web-Text Grundform'. The main content area is divided into two columns. The left column shows the input text, which is a German article snippet about the Motorola Dynatac 8000 mobile phone. The right column displays analysis results under the heading 'Textverständlichkeit' (Text Comprehensibility). This section includes a 'Quantität' (Quantity) sub-section with 'Satzlänge' (Sentence length) and 'Wortlänge' (Word length) metrics, each with a '<< Im Text anzeigen' (Show in text) button and an information icon. Below this are 'Stil' (Style) and 'Wortschatz' (Vocabulary) sections. A 'Web-Tauglichkeit' (Web Suitability) section follows, containing 'Scanbarkeit' (Scannability), 'Interaktivität' (Interactivity), and 'Prägnanz' (Conciseness) metrics.

9.2 PHP

9.2.1 Overview

In most cases are some extra files needed to use a web service with PHP. PHP at it's Version 5 is able to send SOAP Requests, all Versions before need to include either PEAR Classes or other classes which supports Requests via SOAP. Therefore here the "nusoap.php" class is used. It provides a very simple handing in all actuell PHP Versions.

9.2.2 Example

In the official example AJAX is used to show the result on the same page as the WYSIWYG-Editor. That's the reason why no HTML, Body, etc. Tags are used there. If you want to implement the PHP Webservice without AJAX, you simply have to add the Standard HTML Tags and wrap them around the code. Of course you have to edit the form with the content to post and set the "action"-Attribut to the path of this file as well. Sample without AJAX can be found in the "simple_verion" folder.

Code for SOAP Service

```
<?php
require_once('nusoap.php');

//connect to webservice and store the connection in variable client
$serviceURL = 'http://live.lingulab.de.dev.media-access.net/service/liveservice.asmx?WSDL';
$client = new soap_client($serviceURL, true);
$error = $client->getError();
if ($error) {
    echo '<h2>Constructor error</h2><pre>' . $error . '</pre>';
}

// params to login into the webservice
$username = 'YourUsername';
$password = 'YourPassword';
$params = array(
    'userName' => $username,
    'password' => $password
);

// Call to webservice mehthod login
$result = $client->call('Login', array($params));

//check for some errors
if ($client->fault) {
    echo '<h2>Error has occured while Login process.</h2>';
} else {
    $error = $client->getError();
    if ($error) {
        echo '<h2>Error has occured.</h2>';
        echo var_dump($error);
    }
}

if ( get_magic_quotes_gpc() )
    $postedValue = htmlspecialchars( stripslashes( $_POST['text'] ) );
else
    $postedValue = htmlspecialchars( $_POST['text'] );
```

```
//save all posted vars to send them to webservice
$text = $_POST['text'];

$kw1 = $_POST['keyword1'];
$kw2 = isset($_POST['keyword2']) ? $_POST['keyword2'] : '';
$kw3 = isset($_POST['keyword3']) ? $_POST['keyword3'] : '';
$configType = isset($_POST['mode']) ? $_POST['mode'] : '01_web-text_grundform.xml';

$params = array('inputData'=>array('Text' =>
$text,'ConfigurationId'=>$configType,'SearchKeyword1'=>$kw1,'SearchKeyword2'=>$kw2,'SearchKeyword3'=>
$kw3),'authenticationKey'=>$result['LoginResult']);
//call the webservice method "ProcessText"
$action = $client->call('ProcessText', array($params));

$res = $action['ProcessTextResult'];
//show the result
echo $res['MeasureStarsHtml'];
echo 'Messung (0-10): '.$res['Measure'];
echo '<a href="'. $res['LinkOnResultPage']. ">Ergebnissdetails anzeigen</a>';
```

The Result of demo should look like this:

