Technical Description of the Syntactic Tagging web service

Version 1.1

SARL au capital de *53.357 E*. RCS Toulouse B 393 498 324. APE 722C SIRET 393 498 324 00010
5 rue du Moulin Bayard 31000 Toulouse
Tél. 05.61.63.03.74 - Fax. 05.61.63.02.87
Site Internet : http://www.synapse-fr.com - Mail info@synapse-fr.com

Content

1. Modification

Version 1.1

- Unified apikey system.

2. Overview

The aim of this document is to present and explain the Syntactic Tagging Web-service API. It provides a description of the web-service function, and a comprehensive example of consumption.

3. Web-service access

This web-service is available both through a SOAP and a REST access point. This section describes how to connect to one of these two access points.

3.1. SOAP access

The web-service expose a WSDL at this address:

http://api-synapse-dev.azurewebsites.net/syntax/EtiquetteWCF.svc?singleWsdl

This WSDL can be used to automatically generate a proxy for your application.

Manual configuration

If you cannot use the WSDL, or if you prefer to build manually the request, here are the connection parameters for the service:

Service address:
http://api-synapse-dev.azurewebsites.net/syntax/EtiquetteWCF.svc
Service host:
http://api-synapse-dev.azurewebsites.net/syntax
SOAPAction:
http://api-synapse-dev.azurewebsites.net/syntax/IEtiquetteWCF/GetSyntaxAnalysis

The web-service is configured to receive and send XML.

3.2. REST access

In order to provide a convenient way to call the web service from a JQuery, the function is also available at:

http://api-synapse-dev.azurewebsites.net/syntax/EtiquetteWCF.svc/rest/GetSyntaxAnalysis

The section 6 provides an example of how to call the web-service using JQuery and the REST access point.

4. Function description

Synapse_Analysis_Output_SyntaxAnalysis GetSyntaxAnalysis (Synapse_Analysis_Input input);

Description:

The function takes as input a structure containing the credentials identifying the client willing to consume the web-service, the language of the text, and the text to analyse.
The function returns another structure with the text augmented by the detected opinions (as XML annotations), and the possible exception and error codes raised by the service.

Argument:

input is a structure containing the following elements:

- *apikey*: the apikey (as given by Synapse)
- *format*: (optional) desired output format. Default is csv (see description in additional documentation file).
- *lang*: language of the text. Currently, only "*fr*" for French is supported.
- *text*: the text to be processed

XML example for input:

```
<Synapse_Analysis_Input>
    <apikey>apikey</apikey>
    <format>xml</format>
    <lang>fr</lang>
    <text>Nous analysons une phrase simple.</text>
</Synapse_Analysis_Input>
```

Returns:

output is a structure containing the following elements:

- analysis: XML descritption of keywords and concepts (format provided in section 5). Can be null/empty if an error occurred.
- *exceptions*: structure containing
    - *errcode*: the error code. 0 if no error were raised.
    - *message*: comprehensive message describing the problem if an error was raised (i.e. if the errcode is different from 0).

Typical XML output corresponding to the input example (some extra concept and keyword nodes pruned for lisibility):

```
<Synapse_Analysis_Output >
	<analysis>
		N° motN° §	N° phrase	Mot	Lemme	Typegram
	Codegram	Syntagme	Fonction	Num Prop.	Pivot	Type	Prop.
	Sens du mot
	Nous analysons une phrase simple.
		==== DEBUT DE PHRASE ====
	1	1	1	Nous	Nous	PPER1P	Pp1.pn	1	S	1
	analyser	Indépendante
	2	1	1	analysons	analyser	VINDP1P	Vmip1p
	2	V	1	analyser	Indépendante	2
	3	1	1	une	un	DETIFS	Da-fs-i	4	D	1
	analyser	Indépendante
	4	1	1	phrase	phrase	NCFS	Ncfs	4	D	1
	analyser	Indépendante	1
	5	1	1	simple	simple	ADJSIG	Afpfs	4	D	1
	analyser	Indépendante	1
	6	1	1	.	.	PCTFORTE	Yps	-	-	-
	-	-
		===== FIN DE PHRASE =====
	</analysis>
	<exceptions>
		<errcode>0</errcode>
		<message/>
	</exceptions>
</Synapse_Analysis_Output>
```

Error code values (errcode):

- 0: no error.
- 1: wrong credentials.
- 2: expired credentials, user does no longer have the right to consume web-service. Contact Synapse to renew your subscription.
- 3: unrecognized language – analysis returned for the default language.
- 4: no text specified, or unrecognized text format.
- 5: internal error. If problem persist, contact support.
- 6 unrecognized output format – analysis returned in default format.

5. Syntactic tagging CSV Output

This section details the CSV output. More details about the codes used in the output can be provided on demand.

A typical CSV output (field analysis in the output structure) is the following:

```
N° mot	N° §	N° phrase	Mot	Lemme	Typegram	Codegram	Syntagme	Fonction	Num Prop.	Pivot	Type	Prop.	Sens du mot
```

```
Nous analysons une phrase simple.
     ==== DEBUT DE PHRASE ====
1    1    1    Nous Nous PPER1P    Pp1.pn    1    S    1
     analyser  Indépendante
2    1    1    analysons analyser  VINDP1P   Vmip1p    2    V
     1    analyser  Indépendante   2
3    1    1    une  un   DETIFS    Da-fs-i   4    D    1
     analyser  Indépendante
4    1    1    phrase    phrase    NCFS Ncfs 4    D    1
     analyser  Indépendante   1
5    1    1    simple    simple    ADJSIG    Afpfs 4    D    1
     analyser  Indépendante   1
6    1    1    .    .    PCTFORTE  Yps  -    -    -    -    -

     ===== FIN DE PHRASE =====
```

The structure of the text buffer is a tabulated CSV, with sentence delimitation tags. Each column is separated by the character '\t'.

The first line describe each column with a short tag name. The next lines can either be:

- Empty (for lisibility purpose)
- A complete sentence (line 2 of the example)
- A sentence delimitation, `==== DEBUT DE PHRASE ====` for the start of a sentence, `===== FIN DE PHRASE =====` for the end of a sentence.
- A word inside a sentence, with the following columns:
    - Word number
    - Paragraph number
    - Sentence number
    - Word, as in the text
    - Lemma
    - Grammatical type Typegram (cf. additional documentation)
    - Grammatical type 2 (cf. additional documentation, easier to parse for most applications)
    - Groups of the word: one word can belong to several groups ; each number in the group code represent a group.
    - Grammatical function
    - Proposal number

6. Javascript example

This example shows how to make a simple javascript call to the web service using JQuery. JQuery library import is required and not included in this sample.

```
var xmlInput = '<Synapse_Analysis_Input><apikey>' + apikey +
'<\/apikey><format>xml<\/format><lang>fr<\/lang><text>' + text
+ '<\/text><\/Synapse_Analysis_Input>';

var analysis = "";
```

```
$.ajax({
     type : "POST",
     url                  :                  "http://api-synapse-
dev.azurewebsites.net/syntaxanalysis/EtiquetteWCF.svc/rest/Get
SyntaxAnalysis",
     data : xmlInput,
     dataType : 'xml',
     contentType : "text/xml",
     success : parseResponse,
     error : errorHandler
});

function parseResponse(output_analysis) {
     keywords          =          new          XMLSerializer()
.serializeToString(output_analysis);

     $("#outputTxt").setValue(analysis);
}

function errorHandler(XHR, textStatus, errorThrown) {
     alert(errorThrown);
}
```