Newsum WebService Manual

Scify

August 5, 2013

Contents

I NewSum Web Service

The NewSum Web Service has been implemented to provide access openly to the NewSum server. Interfaces have been written for access in three languages, namely java § III,php § IV and python § V. For direct access JSON can be used. The webservice calls return a String formatted using the GSon JSON library for java. The structure of the returned strings can be seen here § VI. **It is stressed that the platform uses *openjdk-6***.

II Functions

- ✓ public String getLinkLabels()
- ✓ public String getCategories(String sUserSources)
- ✓ public String getTopics(String sUserSources, String sCategory)
- ✓ public String getTopicsByKeyword(String sKeyword, String sUserSources)
- ✓ public String getSummary(String sTopicID, String sUserSources)

III NewSumWS calls through java

i Interface

In order to use the NewSum Web Service from a java application you can use the following interface. **Add *NewSumInterface.jar* as a library** to your project and use the following methods.

- ✓ public static LinksData getLinkLabels()
- ✓ public static CategoriesData getCategories(ArrayList<String> alUserSources)
- ✓ public static TopicsData getTopics(ArrayList<String> alUserSources,String sCategory)
- ✓ public static TopicsData getTopicsByKeyword(String sKeyword, ArrayList<String> alUserSources)
- ✓ public static SummaryData getSummary(String sTopicID, ArrayList<String> alUserSources)

ii Examples

In order to run the examples you will first need to **add *NewSumInterfaceJava.jar* to your libraries**. In order for the service to be contacted you will also have to create a file in the project folder (usually in the same folder the src folder is) named ***properties.dat***.

The format of the file ***properties.dat*** is:

- ✓ wsdl:the_link_to_the_wsdl_file
- ✓ namespace:the_namespace
- ✓ soap:the_actual_soap_location_url

Example:

```
wsdl:http://83.212.110.120:8080/NewSumFreeService/NewSumFreeService?wsdl
namespace:http://NewSumFreeService.Server.NewSumServer.scify.org/
soap:http://83.212.110.120:8080/NewSumFreeService/NewSumFreeService
```

The order of the lines in the file matters!

The following tests are arbitrary, not all data is extracted in every example.
For further data manipulation you will need to iterate through the lists!

```
try {
    LinksData links=NewSumInstance.getLinkLabels();
    ArrayList <String> values=links.getLinks(15);
    System.out.println("\nLinkLabels \n");
    for(String each : values){
        System.out.println(each);
    }
    CategoriesData categories=NewSumInstance.getCategories(values);
    System.out.println("\nCategories \n");
    for(String each : categories){
        System.out.println(each);
    }
    TopicsData topics=NewSumInstance.getTopics(values,categories.get(0));
    System.out.println("\nTopics\n");
    ArrayList <String> ids=topics.getTopicIDs();
    for(String each : ids){
        System.out.println(each);
    }
    TopicsData topicskey=NewSumInstance.getTopicByKeyword("Scify",null);
    ArrayList <String> idskey=topicskey.getTopicIDs();
    System.out.println("\nTopics by key \n");
    for(String each : idskey){
        System.out.println(each);
    }
    SummaryData summary=NewSumInstance.getSummary(
      "812cc4cb-af4c-48a0-b318-06d72962885f", values);
    ArrayList <String> snippets=summary.getSummaries();
    System.out.println("\nSummaries \n");
    for(String each: snippets){
        System.out.println(each);
    }
} catch (Exception e) {
    System.out.println(e.getMessage());
}
```

IV NewSumWS calls through php

i Interface

The file **NewSumFreeService.php** is needed in order to use this interface. You also need to include the code:
$require\_once('NewSumFreeService.php');$

✓ public function NewSumFreeService($wsdl)

- Constructor of class NewSumFreeService that extends SoapClient.
- **$wsdl** is of type string and specifies the url of the wsdl file location.
- **Must create instance in order to access member functions.**

✓ public function getLinkLabels()

- Returns LinkLabels containing the urls specifying the sources used as input for summarization.
- LinkLabels is an array of objects that contain 2 members .
  - ◇ member **link** string that contains the url of the source .
  - ◇ member **sourceName** string that contains a label - name for the source .

✓ public function getCategories($userSources)

- Returns Categories that correspond to the userSources selected.
- **$userSources** is of type array string and specifies the user's selected sources. 'All' or null can be used as input if all sources wish to be used as input.
- Categories is an array of strings containing the categories.

✓ public function getTopics($userSources,$category)

- Returns Topics that correspond to the userSources selected and the category specified.
- **$userSources** is of type array string and specifies the user's selected sources. 'All' or null can be used as input if all sources wish to be used as input.
- **$category** is of type string and specifies the user's selected category.
- Topics is an array of objects that contain 4 members.
  - ◇ member **topicID** string that contains the unique id for the topic .
  - ◇ member **topicTitle** string that contains the title for the topic .
  - ◇ member **sourcesNum** integer that corresponds to the number of sources used .
  - ◇ member **date** contains the date the event occured, see § ii.

✓ public function getTopicsByKeyword($keyword,$userSources)

- Searches through Topics and returns those that are relevant to the keyword amongst the selected sources.
- **$keyword** is of type string and specifies the user's selected keyword to search for.
- **$userSources** is of type array string and specifies the user's selected sources. 'All' or null can be used as input if all sources wish to be used as input.
- Topics is an array of objects that contain 4 members.

◇ member **topicID** string that contains the unique id for the topic .

◇ member **topicTitle** string that contains the title for the topic .

◇ member **sourcesNum** integer that corresponds to the number of sources used .

◇ member **date** contains the date the event occured, see § ii.

✓ public function getSummary($topicID,$userSources)

- Creates and returns the Summary specified by the topicID using the user's selected userSources.
- **$topicID** is of type string and specifies the user's selected topicID.
- **$userSources** is of type array string and specifies the user's selected sources.
  'All' or null can be used as input if all sources wish to be used as input.
- Summary is an array of 2 types of arrays of objects.
  Sources, and Snippets each containing the following members.
- **sources** object
  - ◇ member **url** string that contains the url that specifies a source.
  - ◇ member **name** string that specifies a name - label for a source .
- **snippets** object
  - ◇ member **summary** string that contains the summary snippet .
  - ◇ member **sourceUrl** string that contains the url that specifies the source used .
  - ◇ member **sourceName** string that specifies a name - label for the source used .
  - ◇ member **feedUrl** string that specifies a url to the news feed § ii.

ii Examples

Initialize!
First we must call the constructor and initialize variable newsum.

```
$newsum = new NewSumFreeService("insert link to wsdl of NewSum web service here!");
```

Now let's make some tests! Now we should be able to run the following tests.

✓ public String getLinkLabels()

```
$linkLabels=$newsum->getLinkLabels();
echo "<br> link labels! <br><br>";
foreach($linkLabels as $linkLabel){
    echo $linkLabel->sourceName."<br>";
    echo $linkLabel->link."<br>";
}
```

✓ public String getCategories(String sUserSources)

```
echo "<br> categories! <br><br>";
$sources=
array("http://www.koutipandoras.gr/?feed=rss2","http://topontiki.gr/rss");
$categories=$newsum->getCategories($sources);
foreach($categories as $category){
    echo $category."<br>";
}
```

✓ public String getTopics(String sUserSources, String sCategory)

```
echo "<br> topics! <br><br>";
$topics=$newsum->getTopics($sources,$category);
foreach($topics as $topic){
    echo $topic->topicID."<br>";
    echo $topic->topicTitle."<br>";
    echo $topic->sourcesNum."<br>";
    echo "year: ".$topic->date->year."<br>";
    echo "month: ".$topic->date->month."<br>";
    echo "day: ".$topic->date->dayOfMonth."<br>";
    echo "hour: ".$topic->date->hourOfDay."<br>";
    echo "minute: ".$topic->date->minute."<br>";
    echo "second: ".$topic->date->second."<br><br>";
}
```

✓ public String getTopicsByKeyword(String sKeyword, String sUserSources)

```
echo "<br> get topics by keyword! <br><br>";
$keyword="Scify";
$topics=$newsum->getTopicsByKeyword($keyword,null);
foreach($topics as $topic){
    echo $topic->topicID."<br>";
    echo $topic->topicTitle."<br>";
    echo $topic->sourcesNum."<br>";
    echo "year: ".$topic->date->year."<br>";
    echo "month: ".$topic->date->month."<br>";
    echo "day: ".$topic->date->dayOfMonth."<br>";
    echo "hour: ".$topic->date->hourOfDay."<br>";
    echo "minute: ".$topic->date->minute."<br>";
    echo "second: ".$topic->date->second."<br><br>";
}
```

✓ public String getSummary(String sTopicID, String sUserSources)

```
echo "<br> get summaries! <br><br>";
$summaries=$newsum->getSummary($topicID,$sources);
$header= $summaries->sources;
$data= $summaries ->snippets;
echo "<br> summary header <br>";
foreach($header as $sourcetag){
    echo $sourcetag->url."<br>";
    echo $sourcetag->name."<br>";
}
echo "<br> summary data <br>";
foreach($data as $snippet){
    echo $snippet->summary."<br>";
    echo $snippet->sourceUrl."<br>";
    echo $snippet->sourceName."<br>";
    echo $snippet->feedUrl."<br>";
}
```

V NewSumWS calls through python

i Interface

The file **NewSumInterface.py** is needs to be imported as a module in order to use this interface. You also need to create a file named ***properties.dat*** in the same folder as the .py script in order to specify the location of the web service.

The format of the file ***properties.dat*** is:

- ✓ wsdl:the_link_to_the_wsdl_file
- ✓ namespace:the_namespace
- ✓ soap:the_actual_soap_location_url

Example:

```
wsdl:http://83.212.110.120:8080/NewSumFreeService/NewSumFreeService?wsdl
namespace:http://NewSumFreeService.Server.NewSumServer.scify.org/
soap:http://83.212.110.120:8080/NewSumFreeService/NewSumFreeService
```

The order of the lines in the file matters!

- ✓ NewSumInterface()
  Instantiate the class in order to use the methods.
- ✓ getLinkLabels()
  - Returns LinkLabels containing the urls specifying the sources used as input for summarization.
  - LinkLabels is a list of objects that contain 2 members .
    - ◇ object **link** string that contains the url of the source .
    - ◇ object **sourceName** string that contains a label - name for the source .

✓ getCategories(userSources)

- Returns Categories that correspond to the userSources selected.
- **userSources** is a list of strings and specifies the user's selected sources. 'All' or null can be used as input if all sources wish to be used as input.
- Categories is a list of strings containing the categories.

✓ getTopics(userSources,category)

- Returns Topics that correspond to the userSources selected and the category specified.
- **userSources** is a list of strings and specifies the user's selected sources. 'All' or null can be used as input if all sources wish to be used as input.
- **category** is of type string and specifies the user's selected category.
- Topics is an array of objects that contain 4 members.
  - ◇ object **topicID** string that contains the unique id for the topic .
  - ◇ object **topicTitle** string that contains the title for the topic .
  - ◇ object **sourcesNum** integer that corresponds to the number of sources used .
  - ◇ object **date** contains the date the event occured, see § ii.

✓ getTopicsByKeyword(keyword,userSources)

- Searches through Topics and returns those that are relevant to the keyword amongst the selected sources.
- **keyword** is of type string and specifies the user's selected keyword to search for.
- **userSources** is a list of strings that specifies the user's selected sources. 'All' or null can be used as input if all sources wish to be used as input.
- Topics is a list of objects that contain 4 members.

◇ object **topicID** string that contains the unique id for the topic .

◇ object **topicTitle** string that contains the title for the topic .

◇ object **sourcesNum** integer that corresponds to the number of sources used .

◇ object **date** contains the date the event occured, see § ii.

✓ getSummary(topicID,userSources)

- Creates and returns the Summary specified by the topicID using the user's selected userSources.
- **topicID** is a string that specifies the user's selected topicID.
- **userSources** is a list of strings that specifies the user's selected sources.
  'All' or null can be used as input if all sources wish to be used as input.
- Summary is a list of 2 types of lists of objects.
  Sources, and Snippets each containing the following members.
- **sources** object

  ◇ object **url** string that contains the url that specifies a source.

  ◇ object **name** string that specifies a name - label for a source .

- **snippets** object

  ◇ object **summary** string that contains the summary snippet .

  ◇ object **sourceUrl** string that contains the url that specifies the source used .

  ◇ object **sourceName** string that specifies a name - label for the source used .

  ◇ object **feedUrl** string that specifies a url to the news feed § ii.

ii Examples

The following tests are arbitrary, not all data is extracted in every example.
For further data manipulation you will need to iterate through the lists!

```
import NewSumInterface

def main():
    #import logging
    #logging.basicConfig(level=logging.INFO)
    #logging.getLogger('suds.client').setLevel(logging.DEBUG)

    client=NewSumInterface.NewSumInterface()
    #print NewSumInterface().getCategories()
    print "Link labels!\n\n"
    for i in client.getLinkLabels():
        print i['link']
        print i['sourceName']

    sources=[i['link'] for i in client.getLinkLabels()]
    print "Categories!\n\n"

    for i in client.getCategories(sources[1:15]):
        print i

    topics=client.getTopics("A     ",sources)
    print "Topics! \n\n"
    for topic in topics:
        print topic['topicID'], " : ", topic['topicTitle']

    topicsFromKey=client.getTopicsByKeyword("   ",sources)
    print "Topics from Keyword!\n\n"
    for topicK in topicsFromKey:
        print topic['topicID'], " : ", topic['topicTitle']

    summaries=client.getSummary(topic['topicID'],["ALL"])
    print "Summaries! \n\n"
    for snippet in summaries['snippets']:
        print snippet['summary']
    return 0

if __name__ == '__main__':
    main()
```

VI JSON Interface

Needn't be considered if you want to access the webservice through java § III , php § IV or python § V!

i data structs *

◇ public String getLinkLabels()

✓ Returned format= $[\{LinkLabel_1\}, \{LinkLabel_2\}, ..., \{LinkLabel_n\}]$
where $LinkLabel_i = \underbrace{"link"}_{\text{format string}} : \underbrace{"linkString"}_{\text{data}}, \underbrace{"sourceName"}_{\text{format string}} : \underbrace{"sourceNameString"}_{\text{data}}$

returned string example
[{"link":"http://www.mysite.gr/?feed20asdrss2","sourceName":"my site"},
{"link":"http://www.angryBananas.com/rss.xml","sourceName":"AngryBananas"}]

◇ public String getCategories(String sUserSources)

✓ Returned format= $[\underbrace{"category"_1}_{\text{data}}, \underbrace{"category"_2}_{\text{data}}, ..., \underbrace{"category"_n}_{\text{data}}]$

returned string example
["Technology","Science","Sport","Greece","World","SciFY News"]

◇ public String getTopics(String sUserSources, String sCategory)

✓ Returned format= $[\{Topic_1\}, \{Topic_2\}, ..., \{Topic_n\}]$
where $Topic_i = \underbrace{"topicID"}_{\text{format string}} : \underbrace{"topicIDString"}_{\text{data}},$
$Topic_i = \underbrace{"topicTitle"}_{\text{format string}} : \underbrace{"topicTitleString"}_{\text{data}},$
$\underbrace{"date"}_{\text{format string}} : \underbrace{"\{dateString\}"}_{\text{date format } ii}, \underbrace{"sourcesNum"}_{\text{format string}} : \underbrace{sources}_{\text{integer data}}$

returned string example
[{"topicID":"bdasbfe-7326-4251",
"topicTitle":"Cheese is bad for you",
"date":{"year":2013,"month":6,"dayOfMonth":18,
"hourOfDay":21,"minute":43,"second":39},"sourcesNum":5},
{"topicID":"bdasdbfe-7236-4271",
"topicTitle":"Life exists not only on Mars but on Snickers too",
"date": {"year":2012,"month":2,"dayOfMonth":18,
"hourOfDay":21,"minute":45,"second":35},"sourcesNum":3}]

*typically in JSON classes are passed in { } brackets and lists in [ ] brackets

◇ public String getTopicsByKeyword(String sKeyword, String sUserSources)

✓ Returned format= $[\{Topic_1\},\{Topic_2\},...,\{Topic_n\}]$

where $Topic_i = \underbrace{''topicID''}_{\text{format string}} : \underbrace{''topicIDString''}_{\text{data}},$

$\underbrace{''topicTitle''}_{\text{format string}} : \underbrace{''topicTitleString''}_{\text{data}},$

$\underbrace{''date''}_{\text{format string}} : \underbrace{''\{dateString\}''}_{\text{date format } ii}, \underbrace{''sourcesNum''}_{\text{format string}} : \underbrace{sources}_{\text{integer data}}$

returned string example

```
[{"topicID":"bdasbfe-7326-4271", "topicTitle":"Cheese is bad for you",
"date":{"year":2013,"month":6,"dayOfMonth":18,
"hourOfDay":21,"minute":43,"second":39},"sourcesNum":5},
{"topicID":"57a864gf0-6342-46a9", "topicTitle":"Life exists not only on
Mars but on Snickers too",
"date": {"year":2012,"month":2,"dayOfMonth":18,
"hourOfDay":21,"minute":45,"second":35},"sourcesNum":3}]
```

◇ public String getSummary(String sTopicID, String sUserSources)

✓ Returned format= $\{\underbrace{''sources''}_{\text{format string}} : \underbrace{[sources]}_{\text{sources format}}, \underbrace{''snippets''}_{\text{format string}} : \underbrace{[snippets]}_{\text{snippets format}}\}$

- $sources = \{source_1\},\{source_2\},...,\{source_n\}$
- $snippets = \{snippet_1\},\{snippet_2\},...,\{snippet_n\}$
- $source_i = \underbrace{''url''}_{\text{format string}} : \underbrace{''urlString''}_{\text{data}}, \underbrace{''name''}_{\text{format string}} : \underbrace{''nameString''}_{\text{data}}$
- $snippet_i = \underbrace{''summary''}_{\text{format string}} : \underbrace{''summaryString''}_{\text{data}}, \underbrace{''sourceUrl''}_{\text{format string}} : \underbrace{''sourceUrlString''}_{\text{data}},$

$\underbrace{''sourceName''}_{\text{format string}} : \underbrace{''sourceNameString''}_{\text{data}}, \underbrace{''feedUrl''}_{\text{format string}} : \underbrace{''feedUrlString''}_{\text{data}}$

returned string example

```
{"sources":["url":"http://www.scifynews.com","name":"scify",
"url":"http://www.gothamcitynews.com","name":"batman"],
"snippets":["summary":"Scify explores the moon",
"sourceUrl":"http://www.scifynews.com",
"sourceName":"scify","feedUrl":"http://scifynews.com/feed.xml",
"summary":"Batman verifies Scify's lunar exploration project",
"sourceUrl":"http://www.gothamcitynews.com",
"sourceName":"batman","feedUrl":"http://gothamcitynews.com/feed.xml"] }
```

ii date format

dateformat "date":{

- $\underbrace{"year"}_{\texttt{format string}} : \underbrace{"year"}_{\texttt{integer data}}$
- $\underbrace{"month"}_{\texttt{format string}} : \underbrace{"month"}_{\texttt{integer data}}$
- $\underbrace{"dayOfMonth"}_{\texttt{format string}} : \underbrace{"dayOfMonth"}_{\texttt{integer data}}$
- $\underbrace{"hourOfDay"}_{\texttt{format string}} : \underbrace{"hourOfDay"}_{\texttt{integer data}}$
- $\underbrace{"minute"}_{\texttt{format string}} : \underbrace{"minute"}_{\texttt{integer data}}$
- $\underbrace{"second"}_{\texttt{format string}} : \underbrace{"second"}_{\texttt{integer data}}$

}