

# Télécom Saint-Étienne : Master Web-Intelligence

## Web Services

### TP n°3 - 3h00

Enseignant Amro Najjar & Jules Chevalier

## Remarks

- You need to form groups of two students (binôme).
- A binôme should write a report containing the answers of questions and exercises listed in this document and send them to [amro.najjar@telecom-st-etienne.fr](mailto:amro.najjar@telecom-st-etienne.fr).
- **you can write the report in French.**
- **DEADLINE: Mardi 27/01/2015**

Most of today's Webservices provide a RESTful API that allows client **applications** to consume the provided service. Example of these API's include the ones provided by *twitter*, *facebook*, *linkedin* etc.. Although these API are intended to be used by apps, a human-friendly documentation is usually provided. For instance, [here](#) is a the documentation of the twitter API. On the left, you have a column showing the available tasks you can do with this API. Note how almost all of the methods exposed by the API are either GET or POST. Here is one of the big differences between REST and SOAP. This makes REST very easy to learn.

In this TD, I chose twitter API as example. However, what you learn is generic and you can apply it, with some modification on Facebook for instance.

## Objectives

- Write a Twitter RESTful client application that .
  1. Can access your twitter account (e.g. your time-line).
  2. Reads data about your friends (people you follow) and your followers.
  3. Aggregates the data of your friends and generate simple statistical information
  4. Post automatic tweets on your wall.

In order to be able to do these tasks, *twitter* requires you to do a process of authentication. Next section guides you through this process.

## 1 Twitter Authentication: OAuth

For each *twitter* account, there should be at least one *twitter* application based on which we can write our client. If you have a *twitter* account, then you can easily create an application. Otherwise you need to create a twitter account first.

### 1.1 Obtaining the Keys and The Tokens

To create the application do the following:

1. Go to your [twitter app page](#) and create an application.
2. [Here](#) is a short video that shows how to create an application.
3. In the fields **Website** and **Callback URL** you can put whatever website e.g. <http://www.yahoo.fr>. Then click on the button **Create Your Twitter Application**.

4. You will get a page confirming that your application was created.
5. Go to the **permissions** tab and change the permissions granted to your application to **Read, Write and Access direct messages** and click on **Update Settings** button.
6. After the update, go to the **Key and Access Tokens** tab. This tab includes the following authentication data:
  - (a) The Consumer Key (API Key).
  - (b) Consumer Secret (API Secret) ..
  - (c) Scroll down and click on "Create my access token". Other two authentication data are now generated:
  - (d) Access Token.
  - (e) Access Token Secret.

These Five values are essential for your *twitter* client. You will use them in any *twitter* client that interacts with this *twitter* account.

## 1.2 Writing a Simple Java Application with Scribe

**Scribe** is a Java library that facilitates the *twitter* authentication (OAuth). In this subsection we write a small Java application that performs the *twitter* authentication using scribe.

1. Open Eclipse (Standard Eclipse is enough no need to open Eclipse EE).
2. **New ►Java Project**. Name your project **your-lastname-ws-twitter**.
3. Create new package **org.lastname.rest**. Add a new class (called it **Main**) to this package.
4. Add a main method to this class. Add a simple "Hello World" message.

### 1.2.1 Use Maven to Manage Dependencies: Get Scribe

Throughout this TP, we will use Maven to manage the dependencies of our Java project. Maven is a build automation tool used primarily for Java projects. Maven provides a very simple manner to download all the dependencies (.jar files ) you need in you Java project. In order to use Maven, you need to convert your project to a **Maven Project**.

1. Right-click on the project name in Eclipse **Package Explorer ►Configure ►Convert to Maven Project ►Configure ►Finish**. Now your project became a Maven project.
2. Note that a file called **pom.xml** was added to your project. You use this file to manage you maven project. Open this file.
3. After the end of the `</version>` tag add the following dependencies:

```
<dependencies>
  <dependency>
    <groupId>org.scribe</groupId>
    <artifactId>scribe</artifactId>
    <version>1.3.5</version>
  </dependency>
</dependencies>
<repositories>
  <repository>
    <id>scribe-java-mvn-repo</id>
    <url>https://raw.github.com/fernandezpablo85/scribe-java/mvn-repo/</url>
    <snapshots>
      <enabled>true</enabled>
      <updatePolicy>always</updatePolicy>
    </snapshots>
  </repository>
</repositories>
```

4. This means that our project has the **Scribe** as dependency and Maven will go and download it from the repository listed in the code.

5. Right click on the project name in eclipse ►Run As ►Maven Install. Now Maven is fetching the needed .jar files. When you get Maven Build Success you can see how Maven Dependencies was added to your project structure. Inside it you can find the Scribe library [scribe-1.3.5.jar](#).

### 1.2.2 Back to Main Class

Copy the following code inside your main method:

```
OAuthService service = new ServiceBuilder().provider(TwitterApi.SSL.class)
    .apiKey("your-api-code-here")
    .apiSecret("your-api-secret-here").build();

Token requestToken = service.getRequestToken();
Token accessToken= new Token("your-access-token-here", "your-access-token-secret-here");
```

Do not forget to get the these values (api-code, api-secret etc.) from you twitter application settings.

Now, the authentication should be working fine. *twitter* api provides us with a URL to test if the authentication worked correctly. Add the following line to your code.

```
/*-----Code Block Number 2-----*/
OAuthRequest request =
new OAuthRequest(Verb.GET, "https://api.twitter.com/1.1/account/verify_credentials.json");

service.signRequest(accessToken, request);

Response response = request.send();
```

This code means:

- You are using an authenticated request (of the type `OAuthRequest` to communicate with *twitter*).
- Your request uses the `GET` method.
- You apply your request to the resource specified in the URL. The name of the resource is `verify_credentials`. You can see its documentation [here](#).
- The method `signRequest` couples your request with the authentication data.
- The method `send()` sends the request to the *twitter* server and retrieves the response.

**IMPORTANT: In the following questions you will need to write request similar to this one. Copy-pasting it is a good idea to start**

*twitter* sent you a response. Print it:

```
System.out.println(response.getBody());
```

You should get a message with a strange format containing information about your twitter account. Your name, location, friend count.

Take a look again at the [documentation](#). Note that the response is of **JSON** type. Next section introduces JSON<sup>1</sup>.

## 2 Introduction JSON

JSON is an open standard format that uses human-readable text to transmit data objects consisting of attribute value pairs. It is used primarily to transmit data between a server and web application, as an alternative to XML.

---

1. JSON will be also introduced in the CM of next week.

Originally, JSON stood for JavaScript Object Notation since it was derived from Javascript. Yet, it became a *technology-independent* tool to communicate webservice over the Internet. JSON fulfills the same role assumed by XML but it is considered simpler and a *Fat-Free Alternative to XML*. Therefore, more and more RESTful services are adopting JSON as a tool to encode their messages.

## 2.1 A Word About JSON Syntax

We do not need to enter into the details of JSON Syntax. What we need to know is mainly the following three issues:

1. *JSON Object* is JSON main data structure. Unlike Java, in JSON we do not define classes and instantiate them in object. Instead objects are defined directly. An object is defined using curly brackets<sup>2</sup>. Here is an example of a JSON object:

```
{"firstName":"John", "lastName":"Doe"}
```

*firstName* is the attribute name and *John* is its value.

2. *JSON Array*: JSON arrays are written inside square brackets<sup>3</sup>. an array can contain multiple objects. Also, an object can contain arrays as attributes. Here is an example of an array of employees:

```
"employees": [
  {"firstName":"John", "lastName":"Doe"},
  {"firstName":"Anna", "lastName":"Smith"},
  {"firstName":"Peter", "lastName":"Jones"}
]
```

In Java there are special libraries created to help you to parse a JSON response. Next subsection gives you an example.

## 2.2 JSON Parsers

**JSON Simple** is a simple and easy-to-use library to parse JSON messages. In order to include this library as a resource in your project, **add it as a dependency in the Maven project**:

1. Copy& paste the dependency [here](#) to the pom.xml inside the </dependencies> tag.
2. Execute "Maven Install": Right click on the project name in eclipse ►Run As ►Maven Install. Now Maven is fetching the needed .jar files and JSON Simple is added to your project.

To use the JSON library: you need to create a *parser* as follows:

```
/*-----Code Block Number 1-----*/
JSONParser parser = new JSONParser();
```

Then pass the JSON request you want to pass as a parameter to the parser:

```
/*-----Code Block Number 2-----*/
Object objetNormal = parser.parse(responseGet.getBody());
```

When receiving a JSON response from *twitter* print it (System.out.println()) to see how this request is composed (array inside object or objects inside an array etc.). Check whether it is

---

2. Accolades en francais.

3. Crochet en francais.

an array of objects or an object containing arrays. Depending on that you need to make the appropriate casting. For instance the following code casts *objectNormal* to JSON objects and then gets the array that is inside it.

```
/*-----Code Block Number 3-----*/
JSONObject jsonObject = (JSONObject)objectNormal;
JSONArray array = (JSONArray) jsonObject.get("users");
```

Then you can write a for loop to access the JSON objects inside the array as follows:

```
/*-----Code Block Number 4-----*/
for (int i = 0; i < array.size(); i++) {
    JSONObject arrayObject = (JSONObject)array.get(i);
    System.out.println(arrayObject.get("attribute-name"));
}
```

Note that "attribute-name" is a String indicating the name of the attribute inside the JSON Object . Also, if *objectNormal* does not contain an array, you can directly access its attributes using the `get("attribute-name"))` method.

## Questions:

### Question 1: parse the JSON response returned by the **verify-credentials** request

Parse the response message using a JSON Parser. The response included information about your *twitter* account. In order to see its structure you can print it:

```
System.out.println(response.getBody());
```

Or you can check [its documentation page](#) on the *twitter* API. As you can see, it is composed a JSON object containing information about your *twitter* account. Please print the following attributes:

- id\_str
- name
- screen\_name
- followers\_count
- friends\_count
- created\_at
- time\_zone

### Question 2: Retrieving Your followers

Write a new REST request to retrieve the list of your followers. Use the **Get followers** request. Print the following information about your followers:

- name
- screen\_name
- followers\_count
- friends\_count
- created\_at
- time\_zone

## Hints

- A good starting point would be to copy and edit the code used to create the request of `verify_credentials`. It is provided in Section 1.2.2 Code-Block-Number-2.
- As you can see from the [API documentation of this request](#), it returns a JSON Object containing an array of `'users'`. Your JSON parser should take care of that. See The Code Blocks Number 1, 2,3, & 4 in Section 2.2 as an inspiration.

## 2.3 Question 3: Analyzing your Twitter Data

Let us say, just for example, that a *twitter star-phile* is an account that follows famous people in *twitter*. In order to know how star-phile is your account, Calculate the average number of followers of your friends<sup>4</sup>.

### Hint

Use the [GET friends/list](#) request. You will get an array of your friends and you can then see the number of their followers.

## 2.4 Question 4: Update Your Status

Use the [POST statuses/update](#) request to update your status. You should write the following status on your wall:

This is my RESTFUL status Generated at add-date-&-time-here.

### Hints

- Use `new Date().toString()` Java method to get the timestamp (date + time).
- Your status should be passed an argument in the URL to the `POST` method. However, it should be changed to **UTF-8** encoding in order to be accepted in the URL. You can use the `URLEncoder` class in Java which contains a method called `encode()` that can transform your status into URL parameter.

### 2.4.1 Question 5: REST vs SOAP

List the verbs that can be sent as a REST Request. To see what the verb you can take a look to the class `Verb` used in request creation. Please explain the difference between REST and SOAP from this standpoint.

## Go Further: The package Twitter4J

Since not all people like to work with GET, POST, etc. methods (they consider that it is very low level), developers have created new packages in Java that can handle communication with RESTful webservices (e.g. twitter).

[twitter4j](#) is an example. It is a *wrapper* for twitter client. We call it wrapper since it is wrapped over all the needed technology to communicate with twitter. Therefore, it integrates JSON parsing, and OAuth handling, and HTTP method invocation.

For instance, you can update your status easily using something like:

```
Status status = twitter.updateStatus(latestStatus);
```

---

4. Your friends are the people you follow. You need to calculate the average number of their followers.

To update your status. Hence you do need to worry about authentication, nor about GET, POST etc. What actually happens is that twitter4j transforms the `updateStatus()` method call to POST methods etc.

Add a `twitter4j` dependency into your project and repeat the questions above using the methods provided in this library.

## Remarks

- Compress the project as a .zip file, attach it with the answers (pdf file) and send it to the email address mentioned above. **Please put "WS-TD" in the subject (title) of the email.**