Part 1: HTTP, JSON and RestFull API

- HTTP Principles
  - The HTTP
  - The JSON format
    - Use
    - Basical example
  - REST

HTTP Principles

Before any technical details of Cytomine, we will briefly resume the *HTTP* principles, the *REST* architecture and the *JSON* format which will be mainly used to share the datas.

The HTTP

The *HyperText Transfer Protocol* (HTTP) is a client-server communication protocol. He is mainly used by the web browsers which want to exchange datas with servers (HTML pages, images, ...). A HTTP request always contains an adress (URL) and a method. The URL identify the server and the remote resource in which we want to access. The method is used to determine the request type. In this project, we will used the following methods:

- HEAD : ask information about the the resource, without the susmentionned resource itself.
- GET : ask the resource, without modification on it.
- POST : push datas to create a new resource on the server.
- PUT : replace or update an existing resource on the server.
- DELETE : Delete a resource on the server.

The JSON format

The *JavaScript Object Notation* (JSON) is a data format which allow a textual representation of structural information. Despite its Javascript notation, it is language-free and mainly supported by various programming language.

A document written at the JSON format is composed of :

- A list of values.
- A set of key/values pairs.

There exists 3 data types :

- Objects.
- Arrays.
- Some generical values.

Use

The JSON format is mainly used by web interfaces for the following arguments :

- It is simple.
- It is not verbose but is still human readable.
- This format is easy to encode in Javascript, the main language for Web interfaces development.

Basical example

For this example, we will describe a *user* resource of Cytomine into JSON format by this way :

An user at JSON

```
{
    "username": "johndoe",
    "firstname": "john",
    "lastname": "doe",
    "password": "doedoe2011",
    "authorities": [
        "Regular",
        "Administrator"
    ]
}
```

Each time a HTTP request is done on a server and it returns the asked resource into JSON format, these resource can be evaluated in the following way :

Parse a JSON resource

```
{
    /* Evaluation of an USER object contained in the variable "data" */
    var user = eval('('+data+')');
    /* Display username */
    var username = user.username;
    console.log(username);
}
```

In the other way, we can encode a Javascript object at the JSON format like in the following code. Once at the JSON format, the resource can be send to the server.

Encoder un objet Javascript au format JSON

```
{
    /* Evaluation of an USER object contained in the variable "data" */
    var user = eval('('+data+')');
    var user.username = "john2";
    /* We write the user in a JSON string */
    var jsonUSER = JSON.parse(user);
    ...
}
```

REST

Representational State Transfer (REST) is a way to organize an application for distributed systems. REST is not a protocole neither a format and must be seen as a architectural style. A REST application follow some simple rules :

- Each resource must be able to be identified by an URL.
- HTTP has all the needed operations (GET, POST, PUT, DELETE) to manipulate (LIST, ADD, UPDATE, DELETE) resources.
- Each operation is self-sufficient. We don't have state.
- A resource can be representned under different formats (HTML, JSON, XML, Image). There is not a mandatory format.