INFO-H-511: Web Services
TP 1 - HTTP Programming

Lecturer: Stijn Vansummeren
Teaching Assistant: Stefan Eppe
http://cs.ulb.ac.be/public/teaching/infoh511

2013–2014

Clone this lab:
`git clone http://wit-projects.ulb.ac.be/rhodecode/INFO-H-511/Labs/Lab1-exercises`

Part I: Manually Querying the Web

In this exercise, we will query the World Wide Web directly through the HTTP protocol. Recall that the general format of an HTTP 1.1 GET request is as follows[1]:

```
GET /index.html HTTP/1.1<CR><LF>
Host: www.example.com<CR><LF>
[Other headers]<CR><LF>
<CR><LF>
```

Exercise 1.1

In the following dialog, identify

- the lines sent by the client and those sent by the server,
- the HTTP method used,
- the status code with which the server answered,
- the content-type(s) accepted by the client/provided by the server,
- the actual data submitted in the request/answered by the server

```
POST /data/shorten HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Accept: application/json, text/javascript, */*; q=0.01
Referer: http://bitly.com/
Host: bitly.com
Content-Length: 141
Cookie: [...]

url=www.ulb.ac.be&basic_style=1&classic_mode=&rapid_shorten_mode=[...]

HTTP/1.1 200 OK
Server: nginx
```

[1] Complete reference available on the web

```
Content-Type: application/json; charset=UTF-8
Content-Length: 176
Cache-Control: no-cache, no-store, max-age=0, must-revalidate

{"status_code": 200, "data": {"url": "http://bit.ly/8UEV5N", "hash": "8UEV5N", [...]
```

Exercise 1.2

For the following exercise, you can use the `curl` utility and the piping mechanism from a terminal.

```
curl -v -X <method> -H "<header>: <value>" <url>
```

Construct and execute the following HTTP requests:

- Retrieve the home page of this course (`http://cs.ulb.ac.be/public/teaching/infoh511`). Repeat with the HEAD method instead of GET. Explain the difference.
- Retrieve the footer image of that web page ((`http://cs.ulb.ac.be/public/lib/tpl/ulb/images/footer.gif`). What is the Content-Type being served? Display this image to ensure that you received it correctly.
- GET Google in German. Hint: use the 'Accept-Language' header.
- **Supplemental** Request `http://dbpedia.org/resource/Berlin` in both the `application/rdf+xml` and `text/html` media types. Explain the result.

Part II: Interrogating REST-style Web Services

In the first part of this lab, we have explored ways of manually interrogating the web. We turn to a programmatic way to query services over HTTP.

Exercise 1.3

Google provides HTTP APIs to deal with map and address data. In this exercise, we will use Google Geocode to convert a street name to a location, and Google Static Maps to display the corresponding map. A code skeleton is provided in the `MapClient` directory.

- Update the `getLocations` functions to query Google's Geocode [2] service.
- Parse the service response in the `parseGoogleServiceResponse`, and `parseLocation` functions.
- Download a map in the `getMapImagery` function, with Static Maps [3].

Note that, although strongly suggested by Google, an API key is not required for our usage.

[2] `https://developers.google.com/maps/documentation/geocoding/`
[3] `https://developers.google.com/maps/documentation/staticmaps/`