Web Scraper

rachelemello

December 2015

1 Intro

Web scraping is the technique of retrieving data over the network and then refining it for further processing.

2 Purpose

The purpose of the web scraper is to automatically gather information from different sources on the web.
Firstly it retrieves the titles of the movies displayed this week in Göteborg's cinemas, then for each one of these movies it gathers a range of information about it from different sources.

3 Implementation

3.1 General idea

The web scraper is implemented in Java. It mainly uses the Java native classes *URL*, *InputStreamReader* and *Scanner*.
Using *URL* (takes the address of the website) and *InputStreamReader* (reads a stream of bytes and decodes them into characters) we can create a *Scanner* object.
The problem is that what we get in this *Scanner* object is the HTML data that describes the whole website. The HTML data consists of many hundred of lines; the data we are looking for is there, but it is not so easy to find it. Opening the website on the browser we can view the source code of the page (example on Chrome: View -¿ Developer -¿ View Source), this is what is in our *Scanner* object.
What we do now is read through the source code and look for the data we need. Once we find it, we need to identify some unique tag or string appearing before it: in this way we can automatically locate the data we need.
Once we have these things clear, we use the *Scanner* methods *String findInLine(String pattern)*, which searches for a string that matches the pattern argument in the current line from the input (returns *null* if nothing matches),

and *String nextLine()*, which reads the current line until the end and skips to the next one. This way it is possible to reach the exact line in the HTML code where the data we're interested in is written, and then doing some simple string manipulation we isolate just the data itself.

3.2 Specific Methods

4 Challenges faced

4.1 English titles

The movies on *www.sf.se/filmer/?city=goteborg* are listed with their Swedish title. However, we need the English title to gather the other movie information from *www.omdbapi.com.*

At first, I thought the problem could be solved by accessing the SF page of the individual movie where, among other information, "Originaltitel" is mentioned. However, after some testing, I realised that SF does not always include "Originaltitel", it is not reliable.

After some trial and error, the final solution consists in performing a search on IMDb website with the Swedish title of the movie, as I observed that the first result of this search is always the English title of the movie.

4.2 Poster images

Initially, I was taking the link to the movie posters from the *www.omdbapi.com* which provides the IMDb poster link. Unfortunately, when we tested the website from the server we realised that the posters weren't loading. This is because IMDb blocks requests coming from servers (that's why the posters were loading, instead, when testing the website from a localhost).

The solution is to use the url of the posters on SF website, which are scraped at the same time as the Swedish titles. Unfortunately, this solution is not the best because the resolution of these posters is quite small, but I could not find any better solution.