Data Integration
Project report

Antoni Forzpanczyk and Jedrzej Szor

May 29, 2020

Abstract

The report is a conclusion of work done to implement the practical assignment. In this document we will describe the wrappers used, the safety measures taken and the decisions made in order to provide a useful and secure application.

1 Data source analysis

In order to ensure that the movies from the list will be found, we have implemented a system, which uses Wikipedia's internal search engine. In that engine we search for the movie title provided in the text file and add a phrase "film" at the end. Then go to the site pointed to by the first link found by the search engine. If that link proves to not be a movie (i.e. the Movie object cannot be created with data provided by the wrappers) we try the same approach except with the phrase "movie" instead of "film". In case of failure we simply look for the title itself. If all of the above security measures fail, we throw an exception and tell the user that the requested movie could not be found. This system has proved to be quite secure and did not fail with the titles we have tested it with.

2 Wrappers

After saving the html content of the movie's site we extract and save to separate file the infobox table which is a standard on Wikipedia and as such, we can use it. Infobox contains all of the information about the movie and our methods for retrieving title, director, etc. operate on the currentMovieInfobox.html file. We have been basing all of our wrappers on the infobox template accessible here: `https://en.wikipedia.org/wiki/Template:Infobox_film`

2.1 findTitle()

The title is always the first entity in the infobox and as such is always in the first line. In order to retrieve the string we use the following regex:

```
”>[a–zA–Z0–9\s&():.’]+<”
```

After receiving the string we cut the first and last character, i.e. ”¿” and ”¡”.

2.2 findCover()

In order to retrieve the src String we use the following regex:

```
”src=\”\\S*\\b”
```

After receiving the string we cut it to receive only the url and add ”https:” at the beginning.

2.3 findProductionYear()

In order to retrieve the string we use the following regex:

```
”\\b\\d{4}\\b”
```

Then we parse it to int.

2.4 findReleaseDate()

In order to retrieve the string we use the following regexes:

```
”United States”
”\\b\\d{4}−\\d{2}−\\d{2}\\b”
```

We loop through the infobox util the first regex finds match. Then we use the second one to retrieve the date and parse it to Date format. If that approach does not work it means that there is just one release date and because of that it does not have the addition ”United States”. In that case we use just the second regex in order to get the date.

2.5 findCountries()

In order to retrieve the strings we use the following regexes:

```
"Country.*$"
">[A-Za-z\\s]+<"
```

We loop through the infobox util the first regex finds match. Then we loop again using the second one and adding the results to the array of strings until the result matches any of the keywords indicating the next section such as ”Language”, ”Budget”, etc.

2.6 findDirector()

In order to retrieve the string we use the following regexes:

```
"Directed by.*$"
">[A-Za-z\\s]+<"
```

We loop through the infobox util the first regex finds match. Then we loop again using the second one until we receive the result.

2.7 findCast()

In order to retrieve the strings we use the following regexes:

```
"Starring.*$"
">[A-Za-z\\s]+<"
```

We loop through the infobox util the first regex finds match. Then we loop again using the second one and adding the results to the array of strings until the result matches any of the keywords indicating the next section such as ”Music by”, ”Running time”, etc.

2.8 findDistribution()

In order to retrieve the string we use the following regexes:

```
"Distributed by.*$"
">[A-Za-z0-9\\s.()-]+<"
```

We loop through the infobox util the first regex finds match. Then we loop again using the second one until we receive the result.

2.9 findLanguage()

In order to retrieve the strings we use the following regexes:

```
"Language.*$"
">[A-Za-z\\s]+<"
```

We loop through the infobox util the first regex finds match. Then we loop again using the second one and adding the results to the array of strings until the result matches any of the keywords indicating the next section such as "Budget", "Box office", etc.

2.10 findMusic()

In order to retrieve the string we use the following regexes:

```
"Music by.*$"
">[A-Za-z\\s]+<"
```

We loop through the infobox util the first regex finds match. Then we loop again using the second one until we receive the result.

2.11 findBoxOffice()

In order to retrieve the string we use the following regexes:

```
"Box office.*$"
">[0-9\\$A-Za-z\\s.()]+<"
```

We loop through the infobox util the first regex finds match. Then we loop again using the second one until we receive the result. We then cut the $ and parse the result to int.