FLEX

Carlos Gomes A77185 | Helena Martins A82500 | Nuno Silva A78156

(5 de Abril de 2020)

Resumo

Este trabalho prático desenvolvido, visa o desenvolimento de filtros, em *FLEX*, de forma a retirar informação útil consoante o tema em questão.

1 Introdução

O tema sobre o qual nos iremos focar, é o *Transformador Publico2NetLang*, sobre o qual nos é atribuído um excerto de uma página *HTML* contendo comentários de uma notícia publicada no jornal *O Público*. Sendo assim, foram desenvolvidos alguns filtros sobre os quais pretendemos retirar informações dos comentários, tais como o seu identificador, quem publicou, a data da publicação, o comentário em si, entre outros.

Esta informação foi posteriormente exportada num único ficheiro num formato *JSON*, de forma a possibilitar um estudo sócio-linguístico, e facilitar uma possível utilização deste mesmo num outro projeto.

2 Transformador Publico2NetLang

2.1 Enunciado 2.4

Através do website: **https://natura.di.uminho.pt/ jj/pl-20/TP1/pl19tp1.pdf** é nos disponibilizado um excerto de uma notícia publicada pelo jornal *O Público*, sobre o qual irá visar o nosso filtro em FLEX, daí pretendemos retirar:

- Recolher atributos dos comentários;
- Relacionar um comentário com as suas respostas;
- Exportar as informações retiradas para um ficheiro num formato *JSON*;

2.2 Descrição Problema

Tal como foi referido anteriormente, o trabalho prático da unidade curricular de Processamento de Linguagens, baseia-se na análise da estrutura da página de forma a encontrar padrões que nos possibilitarão a captura dos atributos pretendidos. Desta forma e tendo em conta que se trata de uma página *HTML*, poderemos tirar partido das *tags* existentes de forma a retirar o pretendido mais facilmente. Para isto acontecer, foi necessário construir um filtro de texto, recorrendo ao gerador Flex.

2.3 Resolução do Problema

Para a resolução deste problema e para a construção do filtro em questão, foram criadas várias expressões que iremos explorar ao longo do relatório.

2.3.1 Id

Tal como já referido, e de forma a obter o que pretendemos deste ficheiro *HTML*, iremos tirar partido das *tags* existentes. Assim sendo e de uma forma muito breve, verificamos a existência de um elemento, designado *data-comment-id*, nas *tags* ***li***. Deste modo, foi criado a seguinte expressão regular:

```
data\-comment\-id\=\"[a-zA-Z0-9]+[^\"]*
```

Tal como podemos ver, utilizamos o elemento identificador do ID, para especificarmos mais concretamente o que pretendemos. Assim indicamos que queremos apanhar tudo desde a primeira aspa até há última.

2.3.2 User

Para o caso do nome de utilizador, o processo utilizado foi um pouco diferente, isto pois o nome se encontrar entre > e <. Assim sendo criamos uma expressão que capta toda a linha onde se encontra o *username*, desde o início da *tag*.

```
^<a" "href=.*>[a-zA-Z0-9]+[^\<]*
```

Nesta expressão encontramos similaridades relativamente à expressão aplicada no **Id**, contudo, e tal como mencionado anteriormente, extraímos toda a linha, desde o início da tag.

2.3.3 Date

Tal como o Id, verificamos um elemento de uma das tags é *datetime*, assim sendo criamos uma expressão com a mesma linha de pensamento, a uma já referida, e chegamos a seguinte conclusão:

```
datetime\=\"[a-zA-Z0-9\.\-\:]*[^\"]*
```

2.3.4 Timestamp

Relativamente ao timestamp, o processo terá de ser um pouco diferente, isto pois terá de se retirar o datetime e posteriormente convertê-lo num timestamp. Ou seja, uma representação do datetime num número inteiro. Dado que o datetime retirado anteriormente não se encontra no formato requerido, utilizamos um existente numa tag um pouco mais abaixo.

Para captar este novo datetime, criamos uma *COMMENTTHREAD*, ao qual captamos toda a linha onde ele se encontra e posteriormente, aplica-mos a seguinte expressão regular, para capturar o pretendido dessa mesma linha.

Listing 1: Captura da linha

```
^<a.class\=.*\"\>
```

Listing 2: Captura do datetime

```
([0-9]|\.|\:|" ")*
```

Após a obtenção do "novo"datetime, passamos-lhe como argumento a função criada *convertTimestamp*, sobre qual decompõe a string recebida e calcula o timestamp.

2.3.5 CommentText

A obtenção dos textos, foi um grande desafio pois estes não seguiam o mesmo padrão. Assim sendo, seguimos uma abordagem um pouco similar a do *Timestamp*, e criamos um *COMMENTTHREAD* ao qual capturamos tudo entre um até ao início do texto, onde logo seguida aplicamos o filtro para a captura de tudo até se encontrar um <, referindo-se ao encerramento da tag .

Listing 3: Captura da tag até início do texto

```
\<p\>\n[ ]*
```

Listing 4: Captura do texto

```
(\.*|\n+)*[^<]*
```

2.3.6 Likes

Após uma análise sobre a estrutura da página *HTML* disponibilizada, verificamos que não era possível obter o número de likes dos comentários, assim sendo atribui-se o valor de 0 a todos os likes.

2.3.7 HasReplies & Número de Replies

De forma a obter o número de replies, era preciso percorrer o ficheiro todo e só depois escrever no *json* gerado. Porém como nós escrevemos o *json*, a medida que percorremos o HTML, não é possível contabilizar e daí indicar se um comentário tem ou não respostas.

2.3.8 Reputação

Ao longo da obtenção das características solicitadas pelos docentes, verificamos a existência de um possível extra ao qual consideramos relevante expor e obter. Esta característica, apresenta-se como a reputação do utilizador que escreveu o comentário em análise. Assim sendo desenvolvemos o seguinte filtro, criado e baseado em outros previamente desenvolvidos e expostos.

$$\hat{}<\mathrm{span}.*\,\mathrm{title}\,\backslash=\backslash"[\mathrm{A\text{–}Z}]*[\mathrm{a\text{–}z}]*\backslash"$$

2.4 Resultado Final

Tal como referido anteriormente, ao longo da leitura e análise do ficheiro HTML, é criado e escrito um ficheiro *json*, ao qual de seguida apresenta-mos o resultado final desta mesma análise.

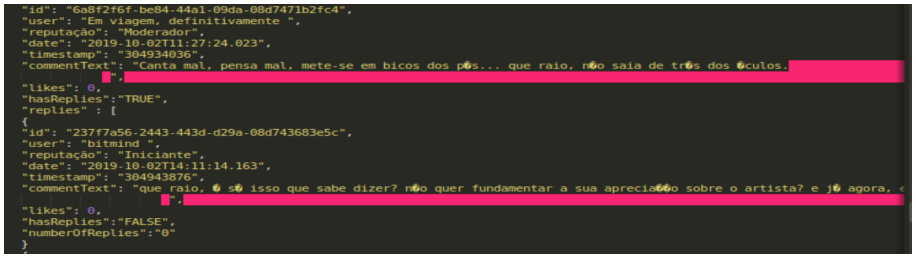

Figura 1: Ficheiro JSON gerado

Como podemos verificar, neste ficheiro, certos caracteres apresentam-se com um "?", isto pois o HTML utilizado, está a usar caracteres de 8bits (nomeadamente CP-1252 que é superset do iso-latin1). Assim ao aplicar, uma das hipóteses de solução por parte dos professores, o ficheiro retornado apresentaria mais desformatações aquelas apresentadas em cima, com isto decidimos continuar com o ficheiro atribuído sem o sujeitar a modificações nos encodings.

3 Conclusão

A realização de um projecto prático na unidade curricular traz sempre benefícios. Ao longo deste projecto, foram desenvolvidas várias expressões regulares, o que acabou por sempre ajudar a consolidar a matéria lecionada até ao momento. Ao mesmo tempo, também dá uma melhor perspectiva do impacto em sistemas mais completos. E claro, ao longo deste projecto, surgiram várias dificuldades, as quais tivemos que aprender a identificar e ultrapassar, o que acaba sempre por ajudar no nosso crescimento.