SHELL INSTAGRAM MEDIA CRAWLER

Aluno: Bruno Tomé

Professor: Walace de Almeida Rodrigues

Instituto Federal de Minas Gerais - Campus Formiga

Outubro de 2016

```
178  # Check if parameters doesn't exists, if doesn't, get user from keyboard
179  if [ $# -eq 0 ]; then
180      echo -n 'Digite seu @usuario no Instagram: '
181      read username
182      username=$(echo $username | sed 's/@//g')
183      echo -n 'Quantidade de fotos para baixar: '
184      countDownloaded=0
185      read maxPics
186  elif [ $# -eq 1 ]; then
187      username=$(echo $1 | sed 's/@//g')
188      maxPics=1000000
189      countDownloaded=0
190  elif [ $# -eq 2 ]; then
191      username=$(echo $1 | sed 's/@//g')
192      maxPics=$2
193      countDownloaded=0
194  else
195      echo 'Quantidade de parâmetros inválida'
196      exit
197  fi
```

“Main" do script

```
199  # If user exists do the wget, if not, remove the files and finish the execution
200  if curl -sSf https://www.instagram.com/$username/media/ > json; then
201
202      rm -rf ./$username
203      mkdir $username
204      cd $username
205      mkdir thumbnail
206      mkdir low_resolution
207      mkdir standard_resolution
208
209      cd ../
210
211      mv json $username
212
213      cd $username
214
215      getPictures
216
217      size=$(checkJSONExists)
218      size=${#size}
```

“Main" do script

```
220     while [ $size -gt 10 ]
221     do
222         if [ $maxPics -gt $countDownloaded ]; then
223             max_id=$(getNextJSON)
224             nextUrl='https://www.instagram.com/'$username'/media/?max_id='$max_id
225             curl -sSf $nextUrl > json
226             getPictures
227             size=$(checkJSONExists)
228             size=${#size}
229         else
230             break
231         fi
232     done
233
234     rm -rf ./json
235 else
236     rm -rf ./json
237     echo 'Usuário inválido'
238 fi
```

“Main" do script

```
28  # Function to call methots to save pictures
29  function getPictures {
30      ##############################################
31      #
32      # Thumbnail
33      #
34      ##############################################
35
36      echo Salvando imagens thumbnail
37      echo " "
38
39      # Enter inside thumbnail directory
40      cd thumbnail
41
42      # Set the type of image to download
43      imageType=thumbnail
44
45      # Call the parseJSON function
46      parseJSON
47
48      # Call the function to download images
49      downloadImages
50
51      # Reset countDownloaded variable
52      countDownloaded=0
```

Função para alternar entre os tamanhos de mídias disponíveis

```
10  # Function to download images by urls in txt file
11  function downloadImages {
12      while read url; do
13          if [ $maxPics -gt $countDownloaded ]; then
14              echo Salvando $url
15              wget -q $url
16              echo ' '
17              ((countDownloaded++))
18          else
19              break
20          fi
21      done < ./urls.txt
22
23      # Remove auxiliar txt
24      rm -rf ./urlsAux.txt
25      rm -rf ./urls.txt
26  }
```

Função para baixar imagens

```
# Function to check if exists more pictures
function checkJSONExists {
    temp=`cat json |
        sed 's/\\\\\//\//g' |
        sed 's/[{}]//g' |
        awk -v k="text" '
            {
                    n=split($0,a,",");
                    for (i=1; i<=n; i++)
                        print a[i]
            }' |
        sed 's/\"\:\"/\|/g' |
        sed 's/[\,]/ /g' |
        sed 's/\"//g' |
        grep -w items`

    echo ${temp}
}
```

Função para checar se existem mais imagens

```
120  # Function to get next JSON
121  function getNextJSON {
122      temp=`cat json |
123          sed 's/\\\\\//\//g' |
124          sed 's/[{}]//g' |
125          awk -v k="text" '
126              {
127                      n=split($0,a,",");
128                      for (i=1; i<=n; i++)
129                          print a[i]
130              }' |
131          sed 's/\"\:\"/\|/g' |
132          sed 's/[\,]/ /g' |
133          sed 's/\"//g' |
134          grep -w id |
135          grep _`
136
137      echo ${temp} | awk -F " " '{print $NF}'
138  }
```

Função para pegar o próximo JSON a partir da ID da última foto

```
# Parse JSON string into a txt containing image links
function parseJSON {
    temp=`cat ../json |
        sed 's/\\\\\\//\//g' |
        sed 's/[{}]//g' |
        awk -v k="text" '
            {
                n=split($0,a,",");
                for (i=1; i<=n; i++)
                    print a[i]
            }' |
        sed 's/\"\:\"/\|/g' |
        sed 's/[\,]/ /g' |
        sed 's/\"//g' |
        grep -w $imageType`
```

Função parsear o JSON no formato solicitado a partir do tamanho de mídia

```
156         echo ${temp} |
157         sed 's/thumbnail: //g' |
158         sed 's/low_resolution: //g' |
159         sed 's/standard_resolution: //g' |
160         sed 's/images: //g' |
161         sed 's/videos: //g' |
162         sed 's/url: //g' |
163         sed 's/\?ig_cache_key[^ .]*\.2//g' |
164         sed 's/\.c / /g' |
165         sed 's/\.l / /g' |
166         awk '
167             {
168               split($0, chars, " ")
169               for (i=1; i <= length($0); i++) {
170                 printf("%s\n", chars[i])
171               }
172             }' > urlsAux.txt
173
174         # Remove blank lines
175         sed '/^$/d' urlsAux.txt > urls.txt
176 }
```

Função parsear o JSON no formato solicitado a partir do tamanho de mídia

```
156         echo ${temp} |
157         sed 's/thumbnail: //g' |
158         sed 's/low_resolution: //g' |
159         sed 's/standard_resolution: //g' |
160         sed 's/images: //g' |
161         sed 's/videos: //g' |
162         sed 's/url: //g' |
163         sed 's/\?ig_cache_key[^ .]*\.2//g' |
164         sed 's/\.c / /g' |
165         sed 's/\.l / /g' |
166         awk '
167             {
168               split($0, chars, " ")
169               for (i=1; i <= length($0); i++) {
170                 printf("%s\n", chars[i])
171               }
172             }' > urlsAux.txt
173
174         # Remove blank lines
175         sed '/^$/d' urlsAux.txt > urls.txt
176 }
```

Função parsear o JSON no formato solicitado a partir do tamanho de mídia

Repositório no GitHub

https://github.com/ibrunotome/Shell-Instagram-Media-Crawler

Referência

Parsear JSON com sed e awk: https://github.com/mauricerenck/code-samples/blob/27e0678796f36144a8dc226b1863455ab989a193/create-changelog/changelog.sh