Tutorial de uso do RSelenium

Daniel Marcelino

07 de junho de 2015

Apresentação

Introdução

O que é o RSelenium?

O **RSelenium** é um pacote no R que permite a manipulação de browsers de internet (Chrome, Firefox, Explorer) por meio de um aplicativo externo em Java. O **Selenium** é um projeto utilizado em diversas linguagens de programação.

Mais informações podem ser encontradas no manual.

`http://cran.r-project.org/web/packages/RSelenium/RSelenium.pdf`

Introdução

Para utilizar o **RSelenium** é necessário instalar o pacote *"RSelenium"* e atualizar o aplicativo externo em **Java** que o pacote utiliza. Felizmente isso pode ser realizado dentro do **R**:

```
## Programação para instalação e atualização do RSelenium
install.packages('RSelenium')
require('RSelenium')
RSelenium::checkForServer()
```

(O pacote só funciona para versões do **R** acima da 3.1.0)

Selenium Server

Caso deseje abrir o browser na mesma máquina, é necessário utilizar um server do Selenium.

```
## Programação para abrir o servidor local do RSelenium
RSelenium::checkForServer()
RSelenium::startServer()
remDr <- remDr <- remoteDriver(remoteServerAddr = "localhost"
                      , port = 4444
                      , browserName = "firefox"
                      )
remDr$open()
```

A programação iniciará um browser (Firefox por padrão) que poderá ser controlado por comandos em sua sessão do R.

Selenium Server

O aplicativo baixado pelo comando checkForServer() pode ser encontrado neste **LINK**, e instalado manualmente na sua pasta de pacotes do R.

É possível modificar o browser utilizado (para Chrome, etc...). Para isso é necessário a instalação de extensões disponíveis na internet, elas não são instaladas por padrão.

No **IPEA** é necessário pedir para atualizarem manualmente o aplicativo pois o downlaod automático é bloqueado. Após o iniciado o browser, algumas vezes é necessário acessar alguma página qualquer manualmente para que os comandos via **R** comecem a funcionar (Não sei o motivo).

Selenium Server

Após iniciado server, para obter informações sobre o servidor e da conexão, basta utilizar a função:

```
## Função retorna status da conexão
remDr$getStatus()
```

Comandos Básicos

Os comandos a seguir são os comandos básicos para navegação utilizando o browser do RSelenium:

```
## Função diz um endereço para o browser acessar
remDr$navigate("http://www.google.com")
## Diz qual a página que o browser está no momento
remDr$getCurrentUrl()
## Retorna para a página anterior do browser
remDr$goBack()
## Avança para a última tela acessada pelo browser
remDr$goForward()
## Carrega novamente a página atual
remDr$refresh()
```

Estrutura Básica

O Modelo de Objeto de Documento (do inglês Document Object Model - DOM) é uma convenção multiplataforma e independente de linguagem para representação e interação com objetos em documentos HTML, XHTML e XML.

Para realizar extrações de páginas com esse formato, é necessário entender como ele é organizado.

Estrutura Básica

Se você acessar a página do Google, verá que o código da página segue esse modelo.
Inspecionando a barra de busca você verá o seguinte código:

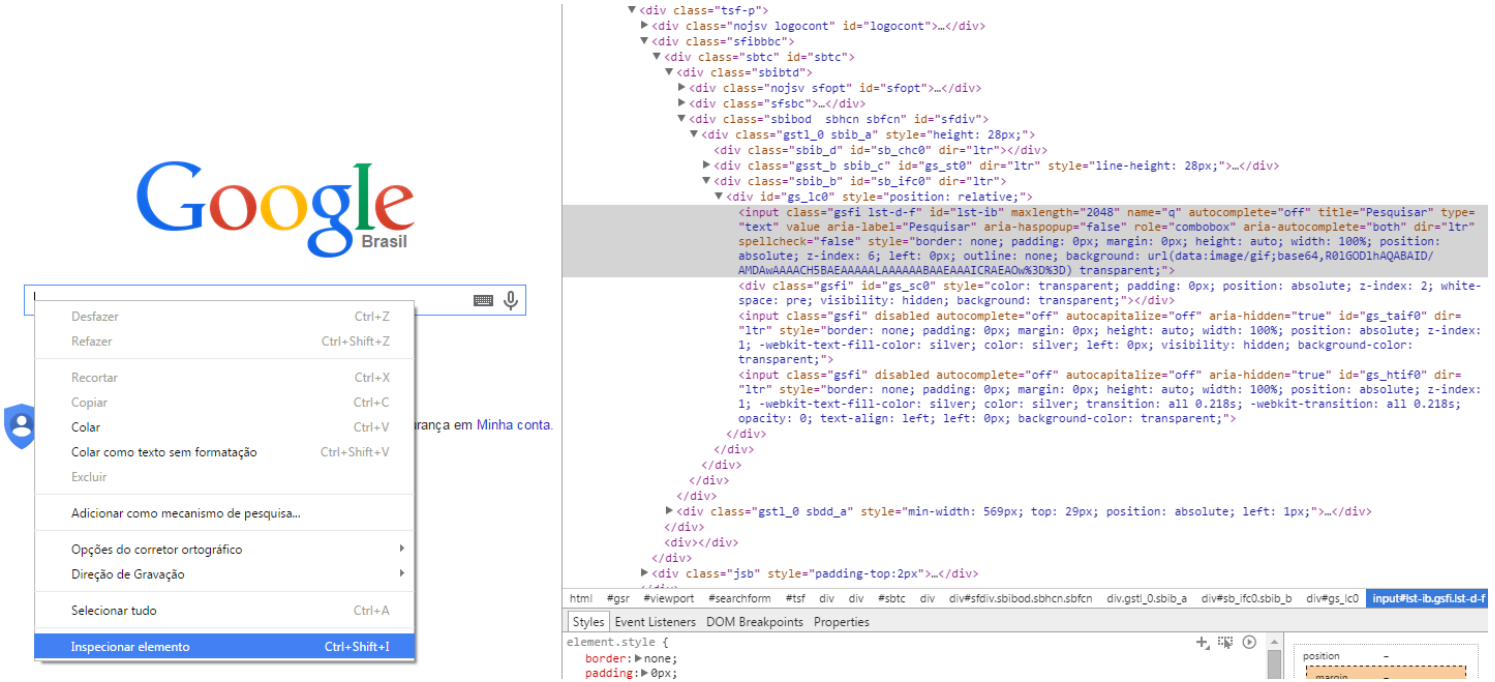

Figura: Código da barra de busca

Estrutura Básica

A forma mais comum de se procurar alguma informação em um **DOM** é utilizando alguma das seguintes estruturas:

- **id**
- **class**
- **css-selector**
- **name**
- **xpath**

O comando no **R** para buscar um objeto com algum desses caminhos é:

```
## Função para buscar um objeto na página
webElem <- remDr$findElement()
```

Estrutura Básica - **id**

Id's são únicos para cada elemento de uma página, então um **id** se refere a apenas um objeto na página. Você pode identificar o **id** de um objeto inspecionando o mesmo e procurando em seu código.

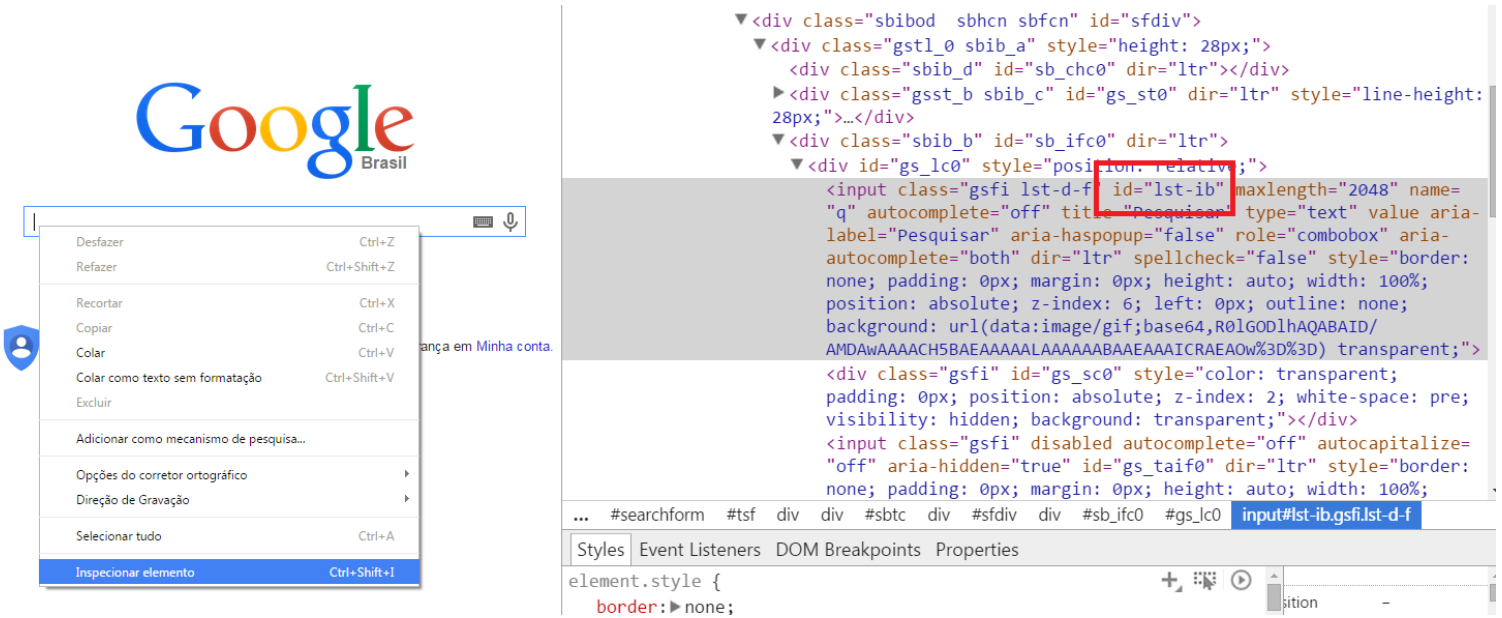

Figura: Código **id**

Estrutura Básica - **id**

O código para buscar o elemento na página seria então:

```
## Função para buscar um objeto na página
webElem <- remDr$findElement(using = "id", "lst-ib")
```

Estrutura Básica - **class**

Você também pode identificar um objeto por sua **class**. Um objeto pode possuir múltiplas classes.

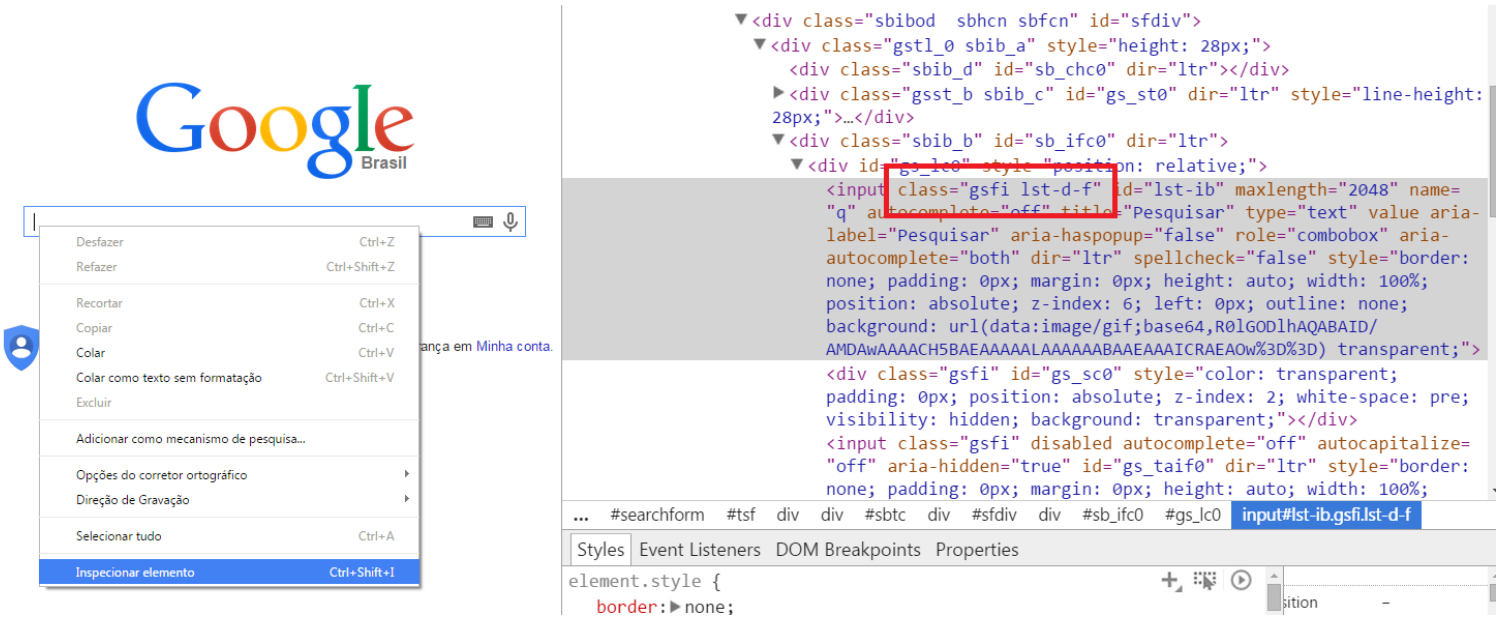

Figura: Código **class**

Estrutura Básica - **class**

O código para buscar o elemento na página seria então:

```
## Função para buscar um objeto na página
webElem <- remDr$findElement(using = "class name", "gsfi")
```

Estrutura Básica - **css-selector**

Os **css-selectors** podem ser usados para identificar objetos com mesmos parentes. "#" é utilizado para caminhos de **id** e "." para caminhos de **class**.

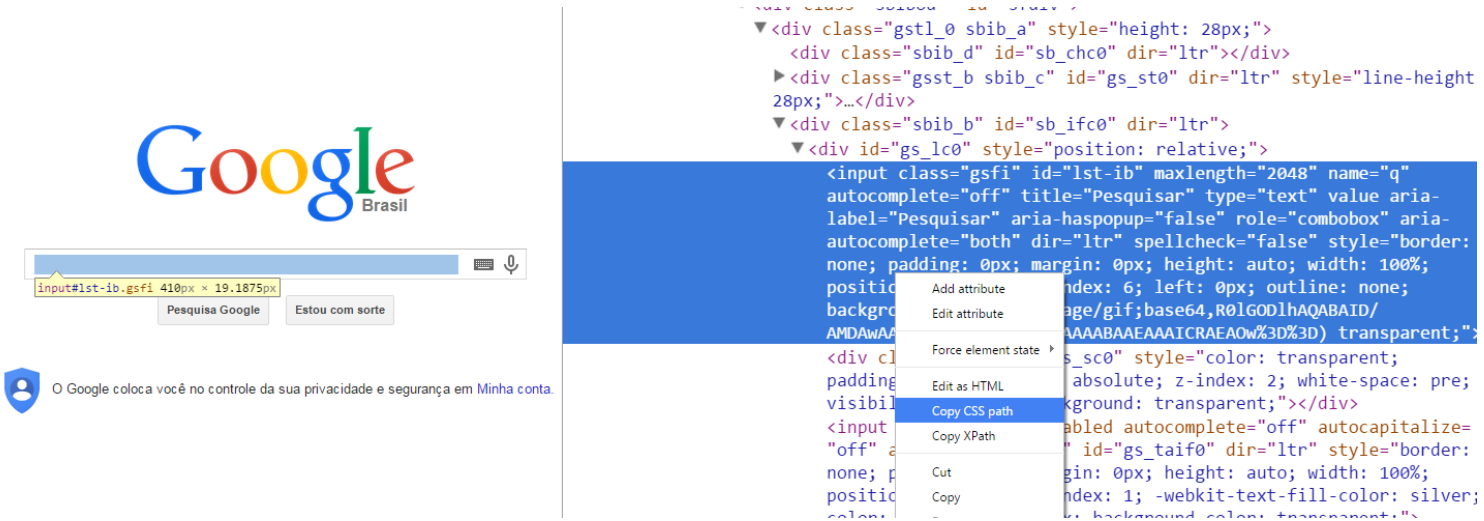

Figura: Código **css-selector**

Estrutura Básica - **css-selector**

O código para buscar o elemento na página seria então:

```
## Função para buscar um objeto na página
webElem <- remDr$findElement(using = "css selector", "#lst-ib")
```

Mais dicas sobre **css-selectors** podem ser encontradas **aqui**.

Estrutura Básica - **name**

Não é muito comum a utilização de **names** na busca de um objeto, pois vários objetos podem possuir o mesmo **name**, dificultando a identificação.

Figura: Código **name**

Estrutura Básica - **name**

O código para buscar o elemento na página seria então:

```
## Função para buscar um objeto na página
webElem <- remDr$findElement(using = "name", "q")
```

Estrutura Básica - **xpath**

O meio mais utilizado para busca de objetos é o **xpath** pois pode ser facilmente obtido e indica diretamente o objeto que se deseja.

Figura: Código **xpath**

Estrutura Básica - **xpath**

O código para buscar o elemento na página seria então:

```
## Função para buscar um objeto na página
webElem <- remDr$findElement(using = "xpath",
    "//*[@id='lst-ib']")
```

Estrutura Básica - Atributos

Caso você identifique o objeto de interesse, é possível verificar seus outros atributos por meio do comando "webElem$getElementAttribute()":

```
## Função para obter os atributos de um objeto
webElem$getElementAttribute("class")
webElem$getElementAttribute("type")
webElem$getElementAttribute("name")
webElem$getElementAttribute("id")
```

Isso é útil caso você identifique o objeto por um atributo mas deseje utilizar outro.

Manipulação da página

Após identificado um objeto de interesse, você pode querer interagir com ele, seja enviando um texto ou selecionando uma opção. O **RSelenium** torna possível essa interação.

O comando "webElem$sendKeysToElement" envia uma ação para seu objeto selecionado:

```
## Função enviar ação
remDr$navigate("http://www.google.com/")
webElem <- remDr$findElement(using = "xpath",
    "//*[@id='lst-ib']")
webElem$sendKeysToElement(list("IPEA"))
```

Esse comando escreverá "IPEA" na barra de texto do **Google**.

Manipulação da página

Você pode desejar enviar ações que não sejam texto, por exemplo, pressionar "Enter". O quadro abaixo mostra a relação de todos os códigos relacionados a uma tecla.

Key	Code
NULL	U+E000
Cancel	U+E001
Help	U+E002
Back space	U+E003
Tab	U+E004
Clear	U+E005
Return[1]	U+E006
Enter[1]	U+E007
Shift	U+E008
Control	U+E009
Alt	U+E00A
Pause	U+E00B
Escape	U+E00C

Key	Code
Space	U+E00D
Pageup	U+E00E
Pagedown	U+E00F
End	U+E010
Home	U+E011
Left arrow	U+E012
Up arrow	U+E013
Right arrow	U+E014
Down arrow	U+E015
Insert	U+E016
Delete	U+E017
Semicolon	U+E018
Equals	U+E019

Key	Code
Numpad 0	U+E01A
Numpad 1	U+E01B
Numpad 2	U+E01C
Numpad 3	U+E01D
Numpad 4	U+E01E
Numpad 5	U+E01F
Numpad 6	U+E020
Numpad 7	U+E021
Numpad 8	U+E022
Numpad 9	U+E023

Key	Code
Multiply	U+E024
Add	U+E025
Separator	U+E026
Subtract	U+E027
Decimal	U+E028
Divide	U+E029

Key	Code
F1	U+E031
F2	U+E032
F3	U+E033
F4	U+E034
F5	U+E035
F6	U+E036
F7	U+E037
F8	U+E038
F9	U+E039
F10	U+E03A
F11	U+E03B
F12	U+E03C
Command/Meta	U+E03D

Figura: Comandos

Manipulação da página

Então para escrever "**IPEA**" e pressionar "**Enter**", é necessário o seguinte comando:

```
## Função enviar ação
remDr$navigate("http://www.google.com/")
webElem <- remDr$findElement(using = "xpath",
    "//*[@id='lst-ib']")
webElem$sendKeysToElement(list("IPEA", "\uE007"))
```

Pois "uE007" é o código associado a tecla "**Enter**"

Identificando diversos elementos

Se quisermos agora verificar os nomes que aparecem no resultado da busca, veja que teremos diversos "blocos", onde cada bloco é um dos resultados da pesquisa.

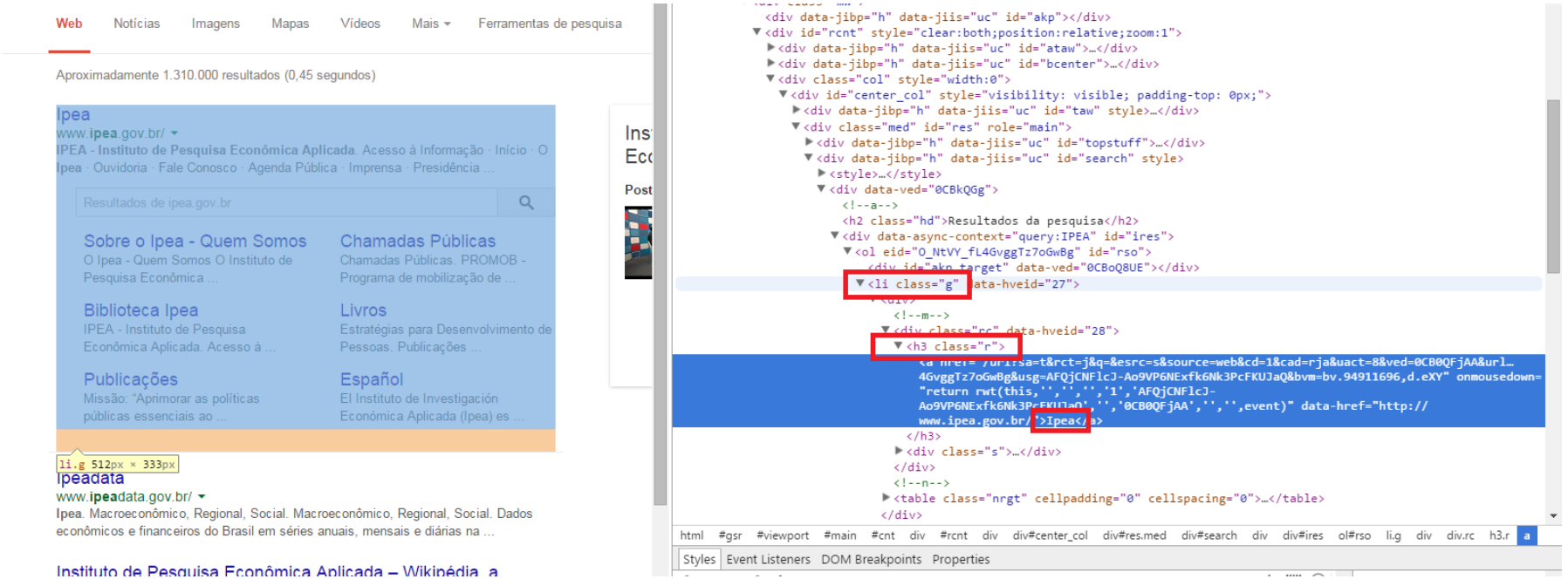

Figura: Nomes

Identificando diversos elementos

Veja que cada bloco está aninhado em <**li class = "g"**> e o texto em <**h3 class = "r"**>, como o link se encontra no **a** é necessário acessá-lo também. Então é necessário acessar cada um desses blocos para obter os nomes que desejamos:

```
## Função identificar textos
webElems <- remDr$findElements(using = 'css selector', "li.g
    h3.r a")
resHeaders <- unlist(lapply(webElems,
    function(x){x$getElementText()}))
resHeaders
```

Veja que como o objeto "webElems" agora é uma lista, pois possui cada bloco separado. Para obtermos o texto que está em cada bloco utilizamos o comando "**getElementText()**" em cada elemento da lista com a função lapply.

Identificando diversos elementos

A programação nos retorna o seguinte resultado:

```
> webElems <- remDr$findElements(using = 'css selector', "li.g h3.r")
> resHeaders <- unlist(lapply(webElems, function(x){x$getElementText()}))
> resHeaders
 [1] "Ipea"
 [2] "Sobre o Ipea - Quem Somos"
 [3] "Chamadas Públicas"
 [4] "Biblioteca Ipea"
 [5] "Livros"
 [6] "Publicações"
 [7] "Español"
 [8] "Ipeadata"
 [9] "Instituto de Pesquisa Econômica Aplicada - Wikipédia, a ..."
[10] "Ipea | Tópicos | EXAME.com"
[11] "Instituto de Pesquisa Econômica Aplicada - YouTube"
[12] "Ipea Instituto de Pesquisa | Facebook"
[13] "Ipea terá guinada progressista com futuro presidente - Carta ..."
[14] "Instituto de Pesquisa Econômica Aplicada"
```

Figura: Resultado da função

Clicar em um elemento

Digamos que nosso interesse seja acessar a página de chamadas públicas do IPEA, então precisamos selecionar o elemento de chamadas públicas e clicar nele com a função "**clickElement()**":

```
## Função clicar
webElem <- webElems[[which(resHeaders == "Chamadas Públicas")]]
webElem$clickElement()
```

Clicar em um elemento

Seguindo a mesma lógica, é simples acessar os links de chamadas públicas do IPEA:

```
#Identificar os nomes da busca no site do IPEA
webElems <- remDr$findElements(using = 'css selector',
    "div#conteudo p a")
resHeaders <- unlist(lapply(webElems,
    function(x){x$getElementText()}))
#Clicar nas chamadas mais recentes
webElem <- webElems[[1]]
webElem$clickElement()
#Selecionar uma chamada
webElems <- remDr$findElements(using = 'css selector',
    "div#conteudo li a")
resHeaders <- unlist(lapply(webElems,
    function(x){x$getElementText()}))
webElem <- webElems[[1]]
webElem$clickElement()
```

Clicar em um elemento

Para fazer download do PDF da chamada é necessário pegar o link e utilizar o comando interno do R "**download.file()**":

```
#Identificar PDF
webElems <- remDr$findElements(using = 'css selector',
    "div#conteudo p a")
resHeaders <- unlist(lapply(webElems,
    function(x){x$getElementAttribute('href')}))
#Download do PDF
download.file(resHeaders[1],'Chamada.pdf',mode='wb')
```

Você pode definir o nome que quiser e o local que deseja salvar o PDF.

Clicar em um elemento

Este foi um tutorial ensinando o básico para utilizar o **RSelenium**, mais informações podem ser encontradas no **help** e neste site que utilizei como base para este tutorial. **Link**

Qualquer dúvida ou correção me mandar um e-mail:
daniel.silva@ipea.gov.br