

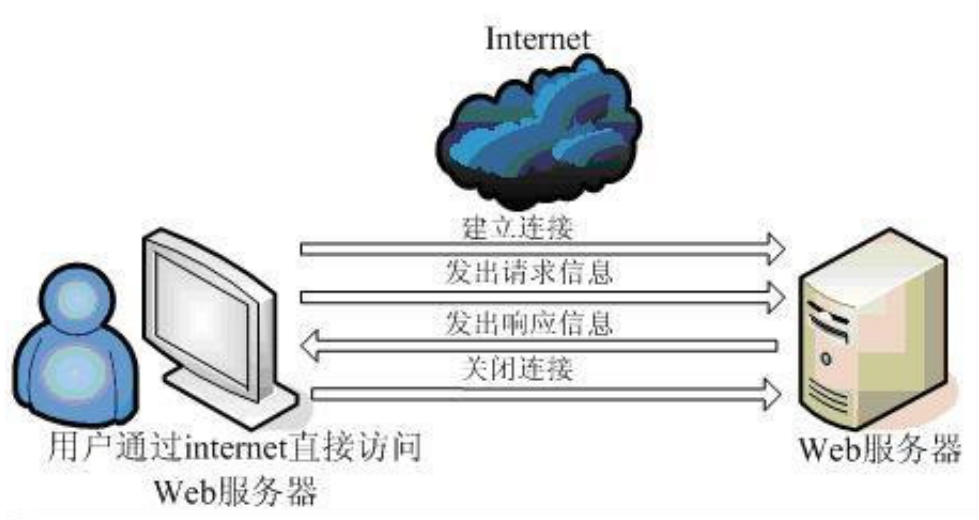
The RCurl package is an R-interface to the libcurl library that provides HTTP facilities. This allows us to download files from Web servers, post forms, use HTTPS(the secure HTTP), use persistent connections, upload files, use binary content, handle redirects, password authentication, etc.

RCurl程序包提供了由R到libcurl库的接口，从而实现HTTP的一些功能。例如，从服务器下载文件，保持连接，上传文件，采用二进制格式读取，句柄重定向，密码认证等。

curl: 利用URL语法在命令行下工作的开源文件传输工具，curl背后的库就是libcurl

## HTTP协议

协议是计算机通信网络中两台计算机之间进行通信所必须共同遵守的规定或规则，超文本传输协议(HTTP)是一种通信协议，它允许将超文本标记语言(HTML)文档从Web服务器传送到客户端的浏览器，目前我们使用的HTTP/1.1版本



## URL详解

◆ 基本格式：schema://host[:port#]/path/.../[?query-string][#anchor]

schema	指定底层使用的协议(例如：http, https, ftp)
host	HTTP服务器的IP地址或者域名
port#	HTTP服务器的默认端口是80，这种情况下端口号可以省略。
path	访问资源的路径
query-string	发送给http服务器的数据
anchor-	锚

基本格式： `schema://host[:port#]/path/.../[?query-string][#anchor]`

scheme：指定底层使用的协议(例如: http, https, ftp)

host：HTTP服务器的IP地址或者域名

port#：HTTP服务器的默认端口是80，这种情况下端口号可以省略

path：访问资源的路径

query-string：发给http服务器的数据

anchor：锚

请求request

## 请求request

◆ 请求行、请求报头、消息正文

METHOD /path - to - resource HTTP/Version-number
Header-Name-1: value
Header-Name-2: value
Optional request body

Method表示请求方法,比如 "GET" , "POST" , "HEAD" , "PUT" 等

Path-to-resource表示请求的资源

Http/version-number 表示HTTP协议的版本号

Method: 表示请求方法, 比如"GET","POST","HEAD","PUT"等

#### Answer / Deepinder

In GET method

- data is submitted as a part of url
- data is visible to the user
- it is not secure but fast and quick

In POST method

- data is submitted as a part of http request
- data is not visible in the url
- it is more secure but slower as compared to GET

Is This Answer Correct ?	208 Yes <input type="radio"/>	9 No <input type="radio"/>	submit
--------------------------	-------------------------------	----------------------------	--------

Path-to-resource: 表示请求的资源

Http/version-number: 表示HTTP协议的版本号

#### 请求报头

### 请求报头

- ◆ Host
- ◆ Accept
- ◆ Accept-encoding
- ◆ Accept-language
- ◆ User-agent
- ◆ Cookie
- ◆ Referer
- ◆ Connection



ATAGURU 炼数成金

DATA GURU 专业数据分析社区

凡七种武器之网络爬虫RCurl 讲师 张瑞仪

Host: 服务器地址

Accept: 浏览器可以接受的媒体类型, text/html

Accept-encoding: 浏览器接收的编码方法, 通常所指的是压缩方法

Accept-language: 浏览器声明自己接受的语言


User-agent: 告诉服务器客户端的操作系统, 浏览器版本

Cookie: 最重要的请求报头的成分, 为了辨识用户身份, 进行session跟踪而储存在用户本地终端上的数据(通常经过加密)

Referer: 跳转页

Connection: 客户端与服务器的连接状态

### 响应response



◆ 状态行、消息报头、响应正文

Http/version-number	status code	message
Header-Name-1: value		
Header-Name-2: value		
Optional Response body		

HTTP/version-number表示HTTP协议的版本号  
status-code 和message表示状态码以及状态信息

ATAGURU 专业数据分析社区

我七种武器之网络爬虫RCurl 讲师 张锐仅

状态行/消息报头/相应正文

HTTP/version-number: 表示HTTP协议的版本号

status-code和message表示状态码以及状态信息

### status-code(状态码)

状态码用来告诉HTTP客户端，HTTP服务器是否产生了预期的Response

HTTP/1.1中定义了5类状态码，状态码由三位数字组成，第一数字定义了响应的类别

1XX: 提示信息，表示请求已经被成功接收，继续处理

2XX: 成功，表示请求已被成功接收，理解，接受

3XX: 重定向，要完成请求必须进行更进一步的处理

4XX: 客户端错误，请求有语法错误或请求无法实现

5XX: 服务器端错误，服务器未能实现合法的请求

消息报头

- ◆ Server
- ◆ Date
- ◆ Last-Modified
- ◆ Content-type
- ◆ Connection
- ◆ X-Powered-By
- ◆ Content-Length
- ◆ Set-Cookie

```
> h=>getURL("http://www.datauru.cn/",headerfunction = h$update)
> h$value()
      Server      Date
"nginx" "Tue, 05 Aug 2014 03:40:08 GMT"
Content-Type      Content-Length
"text/html"      "225561"
Last-Modified      Connection
"Tue, 05 Aug 2014 03:10:01 GMT"      "keep-alive"
      Vary      ETag
"Accept-Encoding"      "\*53e04b09-37119\*"
Accept-Ranges      status
"bytes"      "200"
statusMessage
"OK"
```

Server: 服务器的软件信息, 如nginx

Date: 响应日期

Last-Modified: 上次修改时间

Content-type: 服务器告诉浏览器自己相应的对象类型, text/html

Connection: 服务器和客户端是否保持链接

X-Powered-By: 表示网站是什么技术开发的, 如PHP

Content-Length: 请求返回的字节长度

Set-Cookie: 响应最重要的一个header, 用于把cookie发给相应的浏览器, 每一个写入cookie成一个set-cookie

## HTML

HTML中标签指的是会指定其中包装的文本作为在浏览器分页的标题栏显示的标题, 在实际语法中标签通常以一个<>符号包括起来, 起始标签、内容和终止标签组合起来则成为元素, 如下代码所示:

```
<title> Chinars | 统计之都 </title>
```

起始标签和终止标签都用<>符号包裹, 以便和内容进行区分, 不同的是终止标签会有一个 / 符号以示区别. 一般而言, 每个元素都有一个起始标签和终止标签, 但也不是全部. 例如, <br> 标签表示换行, 则无需 </br> 标签表示终止.

[常用HTML标签如下][<http://www.w3school.com.cn/tags/>]:

标签	描述
<a>	定义锚
<meta>	定义关于HTML文档的元信息
<link>	定义文档与外部资源的关系
<code>	定义计算机代码文本
<p>	定义段落
<h1>-<h6>	定义HTML标题
<div>	定义文档中的节
<span>	定义文档中的节
<form>	定义供用户输入的HTML表单
<script>	定义客户端脚本

标签最重要的一个特征就是属性:

<a href="/chinar/chinar-2013/">第六届中国R语言会议</a>, 标签 <a> 能够把相关的文本(这里指"第六届中国R语言会议")和一个指向另一个地址的超链接关联起来. href="/chinar/chinar-2013/" 这个属性指定锚链接, 浏览器会自动把这类元素转化为带有下划线并且可以点击的样式. 总之, 属性就是让标签能够描述其内容处理方式的选型. 具体属性的作用则根据响应的标签来定.

属性总是处于起始标签的内部, 标签名的右侧, 一个标签拥有多个属性也是常见操作, 多个属性之间用空格分开:

```
<div></div>
```

树状结构

就像文档结构图一样, HTML最大的一个特点就是它呈现出树形结构的样子:

```
<!DOCTYPE html>
<html>
  <head>
    <title> Chinars | 统计之都 </title>
  </head>
  <body>
    第10届中国R会议简介
  </body>
</html>
```

<!DOCTYPE html> 是文档定义类型标签, 忽略这个的话这个例子的第一个元素就是 <html> 元素, 在这个元素的起始和终止标签内, 又有几个标签分别起始和终止: <head><title> 和 <body>. <head> 和 <body> 标签是直接被 <html> 元素包含的, <title> 标签则包含在 <head> 标签内. 一个典型的树形结构就这样被描述.

所谓对HTML的解析, 就是获得有用的HTML文件表征, 运用一个能够理解标记结构特殊含义的程序, 并在某个R的专用数据结构内部重建HTML文本隐含的层次结构, 而不仅仅是读取. 在R语言中, 使用 XML 包中的 htmlParse 函数来解析一个HTML文件得到HTML格式文件(或使用 htmlTreeParse 函数解析HTML文件得到XML格式文件):

```
library(XML)
url <- "http://www.r-datacollection.com/materials/html/JavaScript.html"
exmaple1 <- htmlParse(file = url)
print(exmaple1)
```

## RCurl三大函数

`getURL()` / `getForm()` / `postForm()` 查看请求相应相关信息

### getURL()

这些命令下载一个或多个URLs(aka, URLs). `getURLContent` 作为一个高级别的函数类似 `getURL` 和 `getBinaryURL`, 但是通过查看HTTP header's Content-Type field对应什么样的内容类型被下载. 使用这个来判断bytes是binary, 还是text.

**利用getURL()查看相关信息**

◆ `url.exists()`

◆ `d = debugGatherer()`

`temp <- getURL("http://www.dataguru.cn/", debugfunction=d$update, verbose = TRUE)`

`cat(d$value()[3])` #提交给服务器的头信息

`cat(d$value()[1])` #服务器地址以及端口号

`cat(d$value()[2])` #服务器端返回的头信息

DATAGURU专业数据分析社区

R七种武器之网络爬虫RCurl 讲师 张松海

`url.exists(url="...")` 判断url是否存在

`d <- debugGatherer()` 接受调试信息, 累加来自response的text信息

`temp <- getURL(url="www.baidu.com", debugfunction=d$update, verbose=TRUE)` `d`以update形式保持更新, 会叠加返回信息; `verbose`设置为TRUE时, 保存调试信息, 为FALSE则隐藏对应`d`信息。

`cat(d$value()[3])` 提交个服务器的头信息, 注意使用`cat`输出, 储存和打印的不同(请求报头), 'headerOut'表示出去也就是发出的

```
> d$value()[3]
headerOut
"GET / HTTP/1.1\r\nHost: www.baidu.com\r\nAccept: */*\r\n\r\n"
> cat(d$value()[3])
GET / HTTP/1.1
Host: www.baidu.com
Accept: */*
```



cat(d\$value()[1]) 服务器的地址和端口号

```
> cat(d$value())[1])
Rebuilt URL to: www.baidu.com/
Trying 14.215.177.39...
TCP_NODELAY set
Connected to www.baidu.com (14.215.177.39) port 80 (#0)
Connection #0 to host www.baidu.com left intact
```

cat(d\$value()[2]) 服务器返回的头信息(消息报头), 'headerIn'表示进入也就是获得

```
> cat(d$value()[2])
HTTP/1.1 200 OK
Accept-Ranges: bytes
Cache-Control: no-cache
Connection: Keep-Alive
Content-Length: 14615
Content-Type: text/html
Date: Fri, 11 Oct 2019 10:35:36 GMT
Etag: "5d8b1fec-3917"
Last-Modified: Wed, 25 Sep 2019 08:06:04 GMT
P3p: CP=" OTI DSP COR IVA OUR IND COM "
Pragma: no-cache
Server: BWS/1.1
Set-Cookie: BAIDUID=387B64F61B59D7CA536A3EB22A6C1C46:FG=1; expires=Thu, 31-Dec-37 23:55:55 GMT; max-age=2147483647; path=/; domain=.baidu.com
Set-Cookie: BIDUPSID=387B64F61B59D7CA536A3EB22A6C1C46; expires=Thu, 31-Dec-37 23:55:55 GMT; max-age=2147483647; path=/; domain=.baidu.com
Set-Cookie: PSTM=1570790136; expires=Thu, 31-Dec-37 23:55:55 GMT; max-age=2147483647; path=/; domain=.baidu.com
Vary: Accept-Encoding
X-UA-Compatible: IE=Edge,chrome=1
```

names(d)

```
> names(d)
[1] "update" "value"  "reset"
```

update表示保持更新; value表示对应的值, reset表示清空

d\$reset() 清空d\$value()内容

```
> d$value()
      text      headerIn  headerOut      dataIn      dataOut  sslDataIn  sslDataOut
      ""           ""           ""           ""           ""           ""           ""
```

---

查看服务器返回的头信息, 列表形式



## 利用getUrl()查看相关信息

◆ 查看服务器端返回的头信息

◆ ###列表形式

◆ h = basicHeaderGatherer()

```
txtt=getURL("http://www.dataguru.cn/",headerfunction = h$update)
```

```
names(h$value())
```

```
h$value()
```

```
> h=getURL("http://www.dataguru.cn/",headerfunction = h$update)
> h$value()
      Server      Date
"nginx" "Tue, 05 Aug 2014 03:40:08 GMT"
Content-Type Content-Length
"text/html" "225561"
Last-Modified Connection
"Tue, 05 Aug 2014 03:10:01 GMT" "keep-alive"
Vary ETag
"Accept-Encoding" "\53e04b09-37119/"
Accept-Ranges status
"bytes" "200"
statusMessage
"OK"
```

DATAGURU专业数据分析社区

R七种武器之网络爬虫RCurl 译者 张静仪

basicHeaderGatherer()/basicTextGatherer() 均返回比较详细的，列表/字符串形式的信息

```
h <- basicHeaderGatherer()
```

```
text <- getURL(url="http://www.baidu.com",headerfunction=h$update)
```

```
names(h$value())
```

```
h$value()
```

```
> names(h$value())
[1] "Date" "Server" "Last-Modified" "ETag" "Accept-Ranges" "Content-Length" "Cache-Control"
[8] "Expires" "Connection" "Content-Type" "status" "statusMessage"
> h$value()
      Date      Server      Last-Modified      ETag
"Fri, 11 Oct 2019 10:53:32 GMT" "Apache" "Tue, 12 Jan 2010 13:48:00 GMT" "\"51-47cf7e6ee8400\""
Accept-Ranges Content-Length Cache-Control Expires
"bytes" "81" "max-age=86400" "Sat, 12 Oct 2019 10:53:32 GMT"
Connection Content-Type status statusMessage
"Keep-Alive" "text/html" "200" "OK"
```

查看服务器返回的头信息，字符串形式

```
h <- basicTextGatherer()
```

```
txt <- getURL(url="http://www.baidu.com",headerfunction=h$update)
```

```
names(h$values())
```

```
h$value() / cat(h$value())
```

```
> names(h$value()) ##返回NULL, 说明是字符串形式, 没有列
NULL
> h$value() ##所有内容只是一个字符串
[1] "HTTP/1.1 200 OK\r\nAccept-Ranges: bytes\r\nCache-Control: no-cache\r\nConnection: Keep-Alive\r\nContent-Length: 14615\r\nCon
tent-Type: text/html\r\nDate: Fri, 11 Oct 2019 10:57:49 GMT\r\nEtag: \"5d8b1fec-3917\" \r\nLast-Modified: Wed, 25 Sep 2019 08:06:0
4 GMT\r\nP3p: CP=\" OTI DSP COR IVA OUR IND COM \"\r\nPragma: no-cache\r\nServer: BWS/1.1\r\nSet-Cookie: BAIDUID=0B9E68ECE441F1CF
DC1A60E225378F04;FG=1; expires=Thu, 31-Dec-37 23:55:55 GMT; max-age=2147483647; path=/; domain=.baidu.com\r\nSet-Cookie: BIDUPSID
=0B9E68ECE441F1CFDC1A60E225378F04; expires=Thu, 31-Dec-37 23:55:55 GMT; max-age=2147483647; path=/; domain=.baidu.com\r\nSet-Cook
ie: PSTM=1570791469; expires=Thu, 31-Dec-37 23:55:55 GMT; max-age=2147483647; path=/; domain=.baidu.com\r\nVary: Accept-Encoding\
r\nX-UA-Compatible: IE=Edge,chrome=1\r\n\r\n"
```

查看url请求的访问信息，使用句柄的方式查看，就是操作系统时执行每一操作过程中，会给每一窗口一个唯一的特定的句柄，通过句柄来操纵窗口

## 利用getURL()查看相关信息


◆ 查看curl请求的访问信息

◆ curl = getCurlHandle()

d=getURL("http://www.dataguru.cn/", curl = curl)

getCurlInfo(curl)\$response.code

getCurlInfo(curl)



```
> getCurlInfo(curl)
$effective.url
[1] "http://www.dataguru.cn/"

$response.code
[1] 200

$total.time
[1] 0.39

$namelookup.time
[1] 0.016

$connect.time
[1] 0.063

$pretransfer.time
[1] 0.063

$size.upload
[1] 0
```

R七种武器之网络爬虫RCurl 讲师 张松海

DATAGURU专业数据分析社区

curl <- getCurlHandle() 获取句柄

d <- getURL("http://www.dataguru.cn",curl=curl)

getCurlInfo(curl)\$response.code 获取对应句柄信息

getCurlInfo(curl=curl)

```
> getCurlInfo(curl)$response.code
[1] 200
> getCurlInfo(curl=curl)
$effective.url
[1] "http://www.baidu.com/"

$response.code
[1] 200

$total.time
[1] 0.053265

$namelookup.time
[1] 0.034166
```

设置自己的header(代理服务器或伪装浏览器和操作系统设置header)

```
◆ myheader <- c(
  "User-Agent" = "Mozilla/5.0 (Windows; U; Windows NT 5.1; zh-CN; rv:1.9.1.6) ",
  "Accept" = "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8",
  "Accept-Language" = "en-us",
  "Connection" = "keep-alive",
  "Accept-Charset" = "GB2312,utf-8;q=0.7,*;q=0.7"
)
◆ 如何确认自己的header提交上去了呢？
```

user-Agent: 指定访问服务器所使用的浏览器类型及版本、操作系统及版本、浏览器内核、等信息的标识(百度搜索user-agent): `R.version`

```
> cat(d$value()[3]) ##提交服务器的头信息
GET / HTTP/1.1
Host: www.baidu.com
User-Agent: Mozilla/5.0 (iPhone; U; CPU iPhone OS 4_0_1 like Mac OS X; ja-jp) AppleWebKit/532.9 (KHTML, like Gecko) Version/4.0.5 Mobile/8A306 Safari/6531.22.7
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us
Connection: keep-alive
Accept-Charset: GB2312,utf-8;q=0.7,*;q=0.7
```

### listCurlOptions() 设置其他句柄参数, 170多个之多

verbose: 输出访问的交互信息TRUE

httpheader: 设置访问信息报头

.encoding='UTF-8' 'GBK'

debugfunction,headfunction,curl

.params: 提交的参数组

dirlistonly: 仅读目录, 这个在ftp的网页, 非常好用

followlocation: 支持重定向

maxredirs: 最大重定向次数

### getForm()

提交HTML格式, 提交信息直接附加在url后实现提交, 可实现搜索; 或者使用POST方式, 将名称-值对信息放到请求的主体部分而不是URL行

在必应内搜索'Rcurl'的url为

```
url <- c("http://cn.bing.com/search?q=Rcurl&go=Search&q=Search&qs=n&form=QBRE&sp=-1&pq=rcurl&sc=8-5&sk=&cvid=1C525D5532D544C2AD4507EA1EFE102F")
```

这里q=rcurl 这里就是关键字rcurl,以? 开始, 后续以&为分隔符

getFormParams(query=url) ##查看url的结构和值

```
> getFormParams(query=url) ##查看url的结构和值
      q      go      qs
"Rcurl" "Search" "n"
  form    sp    pq
"QBRE" "-1" "rcurl"
   sc    sk    cvid
"8-5" "" "1C525D5532D544C2AD4507EA1EFE102F"

> names(getFormParams(query=url))
[1] "q" "go" "qs" "form" "sp" "pq" "sc" "sk" "cvid"
```

例如: url <- 'https://cn.bing.com/search?

q=RCurl&qs=n&form=QBRE&sp=-1&pq=rcurl&sc=8-

5&sk=&cvid=16A9B4786C3240CEA5F5E458E2A00180'

```
getFormParams(query=url)
```

```
> getFormParams(query=url)
      q      qs
"RCurl" "n"
  form    sp
"QBRE" "-1"
   pq    sc
"rcurl" "8-5"
   sk    cvid
"" "16A9B4786C3240CEA5F5E458E2A00180"
```

## postForm()

以保密的形式上传我们所要页面提交的信息, 然后获取服务器端返回该页面信息。例如登陆一个页面, 需要账户和密码, 那么我们需要提交账户和密码, 提交的信息要加密, 然后抓取登陆后的页面信息。

base64用于将口令中不兼容的字符转换为网页兼容字符, 不是为了安全隐私目的

## getBinaryURL()

curl部分参数设置

## curl部分参数设置



- ◆ verbose : 输出访问的交互信息
- ◆ httpheader : 设置访问信息报头
- ◆ .encoding = "UTF-8" " GBK"
- ◆ debugfunction, headerfunction, curl
- ◆ .params : 提交的参数组
- ◆ dirlistonly : 仅读取目录  
ftp.wcc.nrcs.usda.gov/data/snow/snow\_course/table/history/idaho/
- ◆ followlocation : 支持重定向  
http://www.sina.com
- ◆ maxredirs : 最大重定向次数

DATAGURU专业数据分析社区

R七种武器之网络爬虫RCurl 讲师 张松海

乱码可设置encoding为'UTF-8'/'GBK'

- dirlistonly: 仅读取目录在ftp很出色

```
url <- "ftp.xxx"
```

```
filename <- getURL(url, dirlistonly=TRUE)
```

这里filename对应获得就是所有的目录名称

- followlocation设置为TRUE可实现自动跳转

```
curl <- getCurlHandle()
```

```
destination <- getURL("http://www.sina.com", curl=curl, followlocation=TRUE)
```

```
getCurlInfo(curl)$response.code
```

- maxredirs定义重定向次数，防止跳入死循环，一般定义为5或3，防止恶性页面循环

**getBinaryURL()下载文件**

## getBinaryURL()下载文件

```
◆ temp<- getBinaryURL(url)
◆ note <- file("hellodata.xls", open = "wb")
◆ writeBin(temp,note)
◆ close(note)
```



```
url <- "http://rfunction.com/code/1201/120103.R"
```

```
temp <- getBinaryURL(url)
```

```
note <- file("120123.R",open="wb")
```

```
writeBin(temp,note)
```

```
close(note)
```

批量下载

## 批量下载文件的一个例子

```
◆ http://rfunction.com/code/1202/
◆ 所需函数 :
◆ getURL()
◆ strsplit()
◆ lapply()
◆ paste()
◆ getBinaryURL()
◆ Sys.sleep()
```

### Index of /code/1202

```
• Parent Directory
• 120201.R
• 120202.R
• 120203.R
• 120204.R
• 120205.R
• 120206.R
• 120207.R
• 120208.R
• 120209.R
• 120210.R
• 120211.R
• 120212.R
• 120213.R
• 120214.R
• 120215.R
• 120216.R
• 120217.R
• 120218.R
• 120219.R
• 120220.R
• 120221.R
• 120222.R
```

查看源代码，利用正则表达，筛选出来文件，批量下载

```

<tr><th colspan="5"><hr></th></tr>
<tr><td valign="top">&nbsp;</td><td><a href="/code/">Parent Directory</a>      </td><td align="r
<tr><td valign="top">&nbsp;</td><td><a href="120201.R">120201.R</a>      </td><td align="r
<tr><td valign="top">&nbsp;</td><td><a href="120202.R">120202.R</a>      </td><td align="r
<tr><td valign="top">&nbsp;</td><td><a href="120203.R">120203.R</a>      </td><td align="r
<tr><td valign="top">&nbsp;</td><td><a href="120204.R">120204.R</a>      </td><td align="r
<tr><td valign="top">&nbsp;</td><td><a href="120205.R">120205.R</a>      </td><td align="r
<tr><td valign="top">&nbsp;</td><td><a href="120206.R">120206.R</a>      </td><td align="r
<tr><td valign="top">&nbsp;</td><td><a href="120207.R">120207.R</a>      </td><td align="r
<tr><td valign="top">&nbsp;</td><td><a href="120208.R">120208.R</a>      </td><td align="r
<tr><td valign="top">&nbsp;</td><td><a href="120209.R">120209.R</a>      </td><td align="r
<tr><td valign="top">&nbsp;</td><td><a href="120210.R">120210.R</a>      </td><td align="r

```

使用函数拆分得到文件名向量，然后设置for循环，依次下载以实现循环下载目的

Sys.sleep(2)防止频繁访问被网站拉黑，休眠2秒

## XML

XML(eXtensible Markup Language), 可拓展标记语言, 首先它和HTML一样, 为一门标记语言, 那它就该有标记语言的全部特征, 这是XML的共性. XML是被设计用来传输和存储数据的, 这和HTML用来显示数据不大一样, 所以又有网络数据交换最流行格式的美誉:

1. 一个XML文档永远以生明该文档的一行代码来开头:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

version="1.0" 用来声明该XML文档的版本号. encoding="ISO-8859-1" 表明编码格式.

2. XML文档必须要有一个根元素, 这个根元素包裹了整个文档, 上例中, 根元素为:

```
<nbaplayer>
...
</nbaplayer>
```

XML是用来传输数据的, 而这个数据通常是放在具体的XML元素中的.

3. 一个XML元素由起始标签和具体内容来定义, 一个元素可以用一个闭合标签来结束, 也可以在起始标签里用一个斜杠(/)来闭合. 元素里可以包含其他元素, 属性, 具体数据等其他内容.

```
<city name="houston"> rockets </city>
```

元素标题: city

起始标签: <city>

终止标签: </city>

数据值: <rockets>

下载表格



◆ [http://www.bioguo.org/AnimalTFDB/BrowseAllTF.php?spe=Mus\\_musculus](http://www.bioguo.org/AnimalTFDB/BrowseAllTF.php?spe=Mus_musculus)

◆ wp <- getURL(url)

◆ doc <- htmlParse(wp, asText = TRUE)

◆ tables <- readHTMLTable(doc)

Stable1	No.	Ensembl ID	Gene ID	Symbol	Family
1	1	ENSMUSG00000029313	17355	Aff1	AF-4
2	2	ENSMUSG00000031189	14266	Aff2	AF-4
3	3	ENSMUSG00000037138	14764	Aff3	AF-4
4	4	ENSMUSG00000049470	93736	Aff4	AF-4
5	5	ENSMUSG00000046532	11835	Ar	Androgen receptor
6	6	ENSMUSG00000021359	21418	Tcfap2a	AF-2
7	7	ENSMUSG00000025927	21419	Tcfap2b	AF-2
8	8	ENSMUSG00000028640	21420	Tcfap2c	AF-2
9	9	ENSMUSG00000042477	332937	Tcfap2e	AF-2

在readHTMLTable(doc)加入参数header= F, 可避免页眉解析错误导致的解析失败。中文界面解析失败, 选择英文界面解析。

或通过getBinaryURL()直接将整个页面下载下来, 保存为"xxx.xls", 抓取中文界面。

## [XPath][<https://www.w3school.com.cn/xpath/index.asp>]

XPath表达式就是选取XML或者HTML文件中节点的方法, 这里的节点, 通常是指XML/HTML文档中的元素。

XPath通过路径表达式(Path Expression)来选择节点信息, 跟文本系统路径一样使用"/"符号来分隔路径:

nodename: 选择该节点的所有子节点

"/": 选择根节点

"//": 选择任意节点

"@": 选择属性

例如:

nbaplayer: 选取nbaplayer元素所有的子节点

/nbaplayer: 选取根节点nbaplayer

//team: 选择所有的team子元素

//@name: 选择所有的name属性值

除此之外, 可以通过给表达式附加一些条件来选择指定的数据, 所有筛选条件可以附在一个[]符号中:

/nbaplayer/team[1]: 选择nbaplayer下第一个team子元素

//city[@name]: 选择带有name属性的team节点

通过网页中路径抓取信息

- ◆ 斜杠 (/) 作为路径内部的分割符。
- ◆ /: 表示选择根节点
- ◆ //: 表示选择任意位置的某个节点
- ◆ @: 表示选择某个属性
- ◆ \* 表示匹配任何元素节点。
- ◆ @\* 表示匹配任何属性值。
- ◆ node() 表示匹配任何类型的节点。



```

▼ <bookstore>
  ▼ <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  ▼ <book category="cooking">

```

节点/属性

### 试一试

根据XPath规则和网页信息，设置对应的选择条件，抓去对应信息

```
url <- "https://www.w3school.com.cn/example/xdom/books.xml"
```

```
doc <- xmlParse(getURL(url))
```

### 添加谓词

```
getNodeSet(doc,"/bookstore/book[1]") ##获得根节点外的第一个节点信息
```

```
getNodeSet(doc,"/bookstore/[position() < 3]") ##获取前两本
```

```
getNodeSet(doc,"/bookstore/book[last()]") ##最后一本
getNodeSet(doc,"//title[@lang]") ##对应属性
```

例如：

◆ <http://t.dianping.com/guangzhou?q=%E7%94%B5%E5%BD%B1>

- ◆ getURL()
- ◆ htmlParse()
- ◆ getNodeSet()
- ◆ sapply()
- ◆ paste()



DATAGURU专业数据分析社区

R七种武器之网络爬虫RCurl 讲师 张松海

打开网页源代码，查看目的信息的规则，设置对应的匹配信息；针对无法读到或读到乱码信息，可以重设header信息，再次尝试；多页面可以参考网页地址更替规律，使用循环函数来读取多页

```

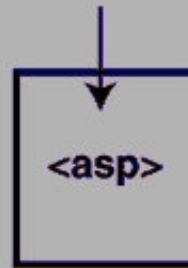
1  # getBinaryURL() 批量下载文件
2  url <- "http://rfunction.com/code/1202/"
3  tmp <- RCurl::getURL(url, httpheader = myheader) # 获取网页
4
5  tmp_files <- strsplit(x=tmp, split="<li><a href='\"'\"'")[[1]]
6  tmp_files1 <- strsplit(tmp_files, split="\"")
7  tmp_files2 <- lapply(X=tmp_files1, function(file) {file[1]})
8  files <- unlist(tmp_files2)
9  files <- files[c(-1, -2)]
10
11 baseURL <- "http://rfunction.com/code/1202/"
12 for(i in 1:length(files)){
13   fullURL <- paste(baseURL, files[i], sep = "")
14   tmp <- getBinaryURL(fullURL)
15   note <- file(paste("1202-", files[i], sep = ""), open = "wb")
16   writeBin(tmp, note)
17   close(note)
18
19   Sys.sleep(2) # 休眠2秒
20 }
    
```

卷

## Using GET

<http://www.somedomain.com/register.asp?name=jobe>

A diagram of a web browser window. At the top, there are five small square icons. Below them is a text input field containing the URL `http://www.somedomain.com/register.asp?name=jobe@electrotank.com`. The main content area of the browser is empty.



## Using POST

<http://www.somedomain.com/register.asp>

A diagram of a web browser window. At the top, there are five small square icons. Below them is a text input field. The main content area of the browser is empty.

HTTP Request

name=jobe&  
email=jobe@  
electrotank.com

