# Getting started with R package `ctrdata` for clinical trial information

*Ralf Herold*

*2019-10-09*

## Contents

## Install package `ctrdata` on a R system

The R Project website (https://www.r-project.org/) provides installers for the R system.

Alternatively, the R system can be used from software products such as R Studio (https://www.rstudio.com/products/RStudio/), which includes an open source integrated development environment (IDE), or Microsoft R Open (https://mran.microsoft.com/open/).

General information on the `ctrdata` package is available here: https://github.com/rfhb/ctrdata.

```
install.packages("ctrdata")
```

The above should install package `ctrdata` into the user's library. If this installation does not succeed, the following sections offer potential solutions.

For using the development version of package `ctrdata`, install from GitHub:

```
# install preparatory package
install.packages(c("devtools", "httr"))
devtools::install_github("rfhb/ctrdata")
```

## Internet access via proxy

Functions in package `ctrdata` that start with `ctr...` require access to trial registers over the internet via `https`. Package `ctrdata` checks and *automatically uses* the proxy that is set under MS Windows in system settings. However, proxy settings need to be set by the user for other operating systems and for authenticating proxies, such as follows:

```
Sys.setenv(https_proxy = "your_proxy.server.domain:8080")
Sys.setenv(https_proxy_user = "userid:password")
```

## Additional installation aspects for MS Windows

On MS Windows, it seems recommended to not use UNC notation (such as `\\server\directory`) for specifying the user's library location:

```
.libPaths("D:/my/directory/")
```

As noted in the README for package `ctrdata`, on MS Windows the cygwin environment has to be installed, into the local directory `c:\cygwin`. The applications php, bash, perl, cat and sed in the cygwin environment are required for function `ctrLoadQueryIntoDb()` of package `ctrdata` (other functions in the package do not have this requirement). The installation of a minimal cygwin environment on MS Windows can be done from package `ctrdata` as follows:

```
ctrdata::installCygwinWindowsDoInstall()
```

If internet accces has to be via a proxy and this was not correctly detected automatically, it can be specified:

```
ctrdata::installCygwinWindowsDoInstall(proxy = "proxy.server.domain:8080")
```

Users who want or need to install cygwin manually can download the setup executable from here. In a MS Windows command window or Powershell window, use the following command line. The parameters are explained here.

```
setup-x86_64.exe --no-admin --quiet-mode --verbose --upgrade-also --root c:/cygwin
--site http://www.mirrorservice.org/sites/sourceware.org/pub/cygwin/
--packages perl,php-jsonc,php-simplexml
```

## Database

At this time, a RSQLite or a remote or a local MongoDB database can be used with the package `ctrdata`. Suggested installation instructions for a local MongoDB server are here; a remote MongoDB database server is accessible here.

## Attach package `ctrdata`

```
library(ctrdata)
```

## Open register's advanced search page in browser

These functions open the browser, where the user can start searching for trials of interest.

```
# Please review and respect register copyrights:
ctrOpenSearchPagesInBrowser(
  copyright = TRUE
)

# Open browser with example search:
ctrOpenSearchPagesInBrowser(
  input = "cancer&age=under-18",
  register = "EUCTR"
)
```

## Click search parameters and execute search in browser

Refine the search until the trials of interest are listed in the browser. Currently, the total number of trials that can be retrieved with package `ctrdata` is intentionally limited to 5000 (CTGOV).

## Copy address from browser address bar to clipboard

Using operating system functions.

## Get address from clipboard

The next steps are executed in the R environment:

```
# q <- "https://www.clinicaltrialsregister.eu/ctr-search/search?
# query=cancer&age=under-18&status=completed&phase=phase-one"

q <- ctrGetQueryUrlFromBrowser()
# Found search query from EUCTR.

q
#                    query-term query-register
# 1 query=cancer&age=under-18            EUCTR

# Open browser with this query
# Note the register needs to be specified
# when it cannot be deduced from the query
ctrOpenSearchPagesInBrowser(
  input = q
)
```

## Retrieve protocol-related information, transform, save to database, check

```
# Connect to a database and chose a table / collection
db <- nodbi::src_sqlite(
  dbname = "sqlite_file.sql",
  collection = "test"
)

# Count number of records
ctrLoadQueryIntoDb(
  queryterm = q,
  only.count = TRUE,
  con = db
)

# Use search q that was defined in previous step:
ctrLoadQueryIntoDb(
  queryterm = q,
  con = db
)

# With file-base SQLite, takes 5 minutes for 1000 records.
# Speed is higher when using MongoDB (or memory-based SQLite).

# Show which queries have been downloaded into the database so far
dbQueryHistory(con = db)
#      query-timestamp query-register query-records                query-term
# 1 2016-01-13 10:51:56          CTGOV         5233 type=Intr&cond=cancer&age=0
# 2 2019-08-12 18:54:40          EUCTR         1838        cancer&age=under-18
```

## Analyse information on clinical trials

```
# find names of fields of interest in database:
dbFindFields(
  namepart = "status",
  con = db
)
# Finding fields on server (may take some time)
# Field names cached for this session.
# [1] "x5_trial_status"
# [2] "b1_sponsor.b31_and_b32_status_of_the_sponsor"
# [3] "p_end_of_trial_status"

# Get all records that have values in all specified fields.
# Note that b31_... is a field within the array b1_...

# NOTE for SQLite:
# For SQLite, but NOT for MongoDB, the element b1_... at the
# root has to be retrieved and only from the nested list in
# this element, the sub-element b31_and_b32_status... can
# then be extracted, using the helpfer function below.
result <- dbGetFieldsIntoDf(
  fields = c("b1_sponsor",
```

```r
                "p_end_of_trial_status"),
    con = db
)

# Show nested list in the element
str(result[1, "b1_sponsor"])
# List of 1
#  $ :'data.frame': 1 obs. of  4 variables:
#   ..$ _b1_sponsor                  : chr "1"
#   ..$ b11_name_of_sponsor          : chr "Wyeth Research
#   ..$ b134_country                 : chr "United States"
#   ..$ b31_and_b32_status_of_the_sponsor: chr "Commercial"

# Helper function
getKeyNestedList <- function(thelist, thekey) {

  # recursive find
  rmatch <- function(x, name) {

    pos <- match(name, names(x))
    if (!is.na(pos)) return(x[[pos]])

    for (element in x) {
      if (any(c("data.frame", "list") %in% class(element))) {
        out <- Recall(element, name)
        if (!is.null(out)) return(out)
      }
    }
  }

  # get key from any level, deduplicate, remove NA,
  # collapse if more than one value per list item
  sapply(thelist, function(x) {
    out <- rmatch(x = x, name = thekey)
    out <- na.omit(out)
    out <- unique(out)
    out <- paste0(out, collapse = " / ")
    out
  })
}

# Extract sought element from root element
result$b31_and_b32_status_of_the_sponsor <-
  getKeyNestedList(
    thelist = result$b1_sponsor,
    thekey = "b31_and_b32_status_of_the_sponsor"
  )

# Tabulate the status of the clinical trial
with(result,
     table(
       "Status"      = p_end_of_trial_status,
       "Sponsor type" = b31_and_b32_status_of_the_sponsor)
```

```
)
#                       Sponsor type
# Status                   Commercial Non-Commercial
#   Completed          1       397              86
#   Not Authorised     1         5               1
#   Ongoing            1       508             431
#   Prematurely Ended  0        82              21
#   Restarted          0         7               3
#   Suspended by CA    0         1               0
#   Temporarily Halted 0         9               7
```