go.data

Go.Data SOP

Retrieving your data collections from API for advanced analysis

Goal of this document:

This SOP will briefly outline the instructions for obtaining your data directly from your Go.Data instance application programming interface (API). This process will provide you with cleaned, flattened excel files for advanced analysis in a program of your choice. Although there are multiple ways to retrieve the data collections including installing and connecting directly to the MongoDB database on your machine, this SOP outlines how to do this simply using the open-source software R – where advanced R skills are not required.

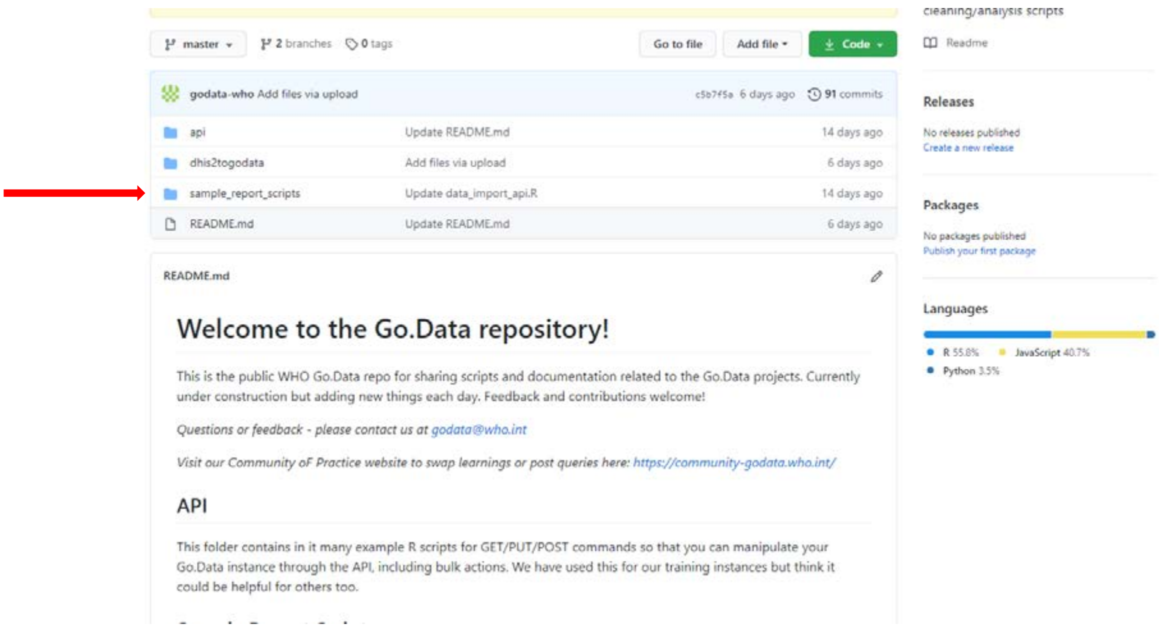
Requirements:

- R (version 3.6.1 or higher is recommended) – download here: https://www.r-project.org/
- Valid Go.Data login credentials for the instance you are accessing, with permissions to view data you are trying to extract
- Folder directory with scripts downloaded onto your computer, accessed from Go.Data Github repository: https://github.com/godata-who/godata/tree/master/sample_report_scripts

For questions please email godata@who.int

Step 1. Replicate folder directory to your local machine

Navigate to godata Github repository. Copy the Go.Data analytics folder hierarchy (labeled **sample_report_scripts**) and all its contents onto your local ://C drive to ensure that there are no issues with sourcing scripts and saving outputs to the proper locations when you are in your R session.



godata-who Add files via upload c5b7f5a 6 days ago 91 commits

api	Update README.md	14 days ago
dhis2togodata	Add files via upload	6 days ago
sample_report_scripts	Update data_import_api.R	14 days ago
README.md	Update README.md	6 days ago

README.md

Welcome to the Go.Data repository!

This is the public WHO Go.Data repo for sharing scripts and documentation related to the Go.Data projects. Currently under construction but adding new things each day. Feedback and contributions welcome!

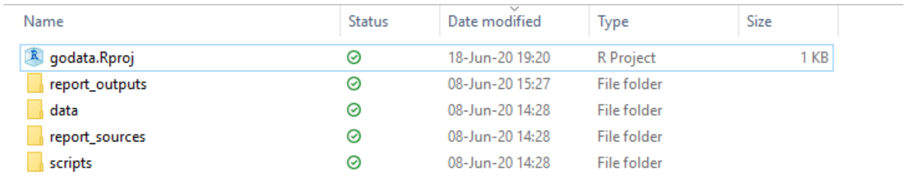
Questions or feedback - please contact us at godata@who.int

Visit our Community oF Practice website to swap learnings or post queries here: https://community-godata.who.int/

API

This folder contains in it many example R scripts for GET/PUT/POST commands so that you can manipulate your Go.Data instance through the API, including bulk actions. We have used this for our training instances but think it could be helpful for others too.

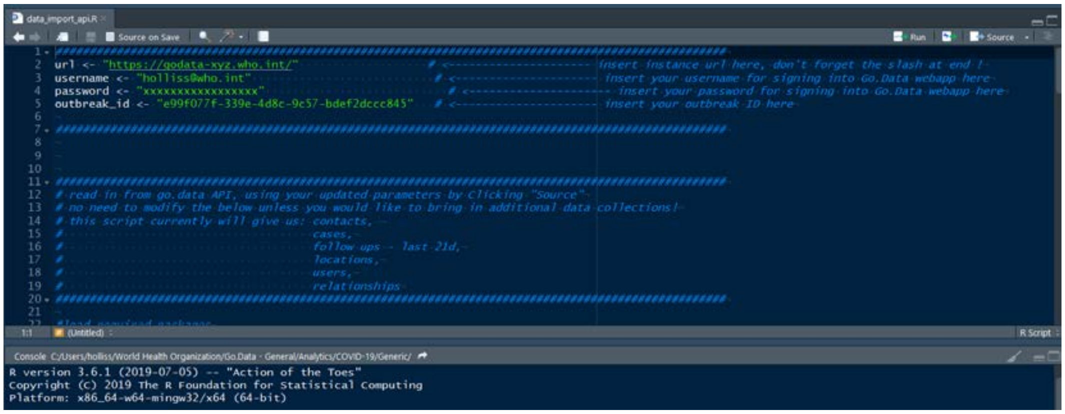
The folder hierarchy contains a **godata.Rproj** file from where you can double-click to start your R session, **report_sources** folder that contains R scripts necessary to execute, **report_outputs** folder where your resulting analysis reports will be saved to if you choose to run analyses, **data** folder where your cleaned data will be saved to, and **scripts** folder that has additional functions or scripts that the **report_sources** are sourcing from but that you do **not** need to run separately. See below for what your high level folder structure should look like.



Name	Status	Date modified	Type	Size
godata.Rproj	✓	18-Jun-20 19:20	R Project	1 KB
report_outputs	✓	08-Jun-20 15:27	File folder	
data	✓	08-Jun-20 14:28	File folder	
report_sources	✓	08-Jun-20 14:28	File folder	
scripts	✓	08-Jun-20 14:28	File folder	

Step 2. Run data import script with your Go.Data credentials

Open up R project by double-clicking on **godata.Rroj**. Navigate to **01_data_import_api.R** in the **report_sources** folder and open it in your R console. At the top of the script, fill in the appropriate **URL**, your Go.Data **username** and **password**, and **outbreak_id** of interest.

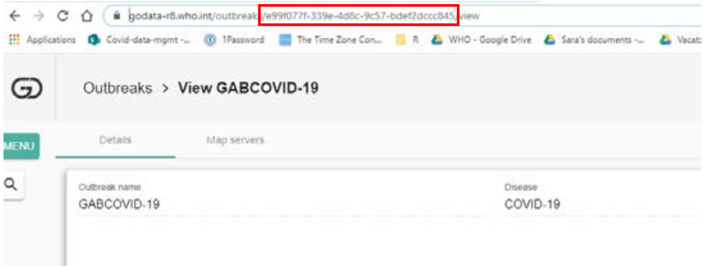


```
####################################################################################
url <- "https://godata-xyz.who.int/"                # <-------------------- insert instance url here, don't forget the slash at end !
username <- "holliss@who.int"                       # <-------------------- insert your username for signing into Go.Data webapp here
password <- "xxxxxxxxxxxxxxxxx"                       # <-------------------- insert your password for signing into Go.Data webapp here
outbreak_id <- "e99f077f-339e-4d8c-9c57-bdef2dccc845"  # <-------------------- insert your outbreak ID here

####################################################################################



####################################################################################
# read in from go.data API, using your updated parameters by clicking "Source"
# no need to modify the below unless you would like to bring in additional data collections!
# this script currently will give us: contacts,
#                                     cases,
#                                     follow ups - last 21d,
#                                     locations,
#                                     users,
#                                     relationships
####################################################################################
```

Console C:/Users/holliss/World Health Organization/Go.Data - General/Analytics/COVID-19/Generic/

```
R version 3.6.1 (2019-07-05) -- "Action of the Toes"
Copyright (C) 2019 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)
```

TIP: In order to obtain your outbreak ID, navigate to View Outbreak in Go.Data and you can find it in the URL.



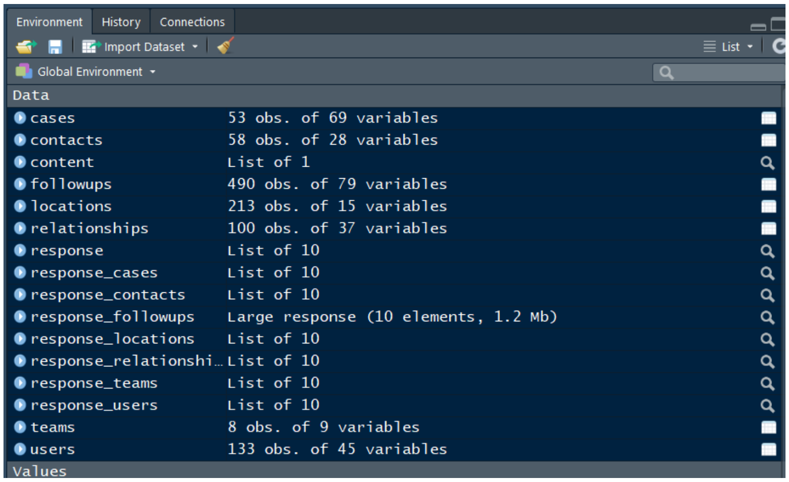
godata-r8.who.int/outbreaks/e99f077f-339e-4d8c-9c57-bdef2dccc845/view

Outbreaks > View GABCOVID-19

Outbreak name
GABCOVID-19

Disease
COVID-19

At the end of this script, you should have created several data frames in your R global environment that will be used in subsequent cleaning scripts, including: **cases; contacts; followups; locations; relationships; teams; users**



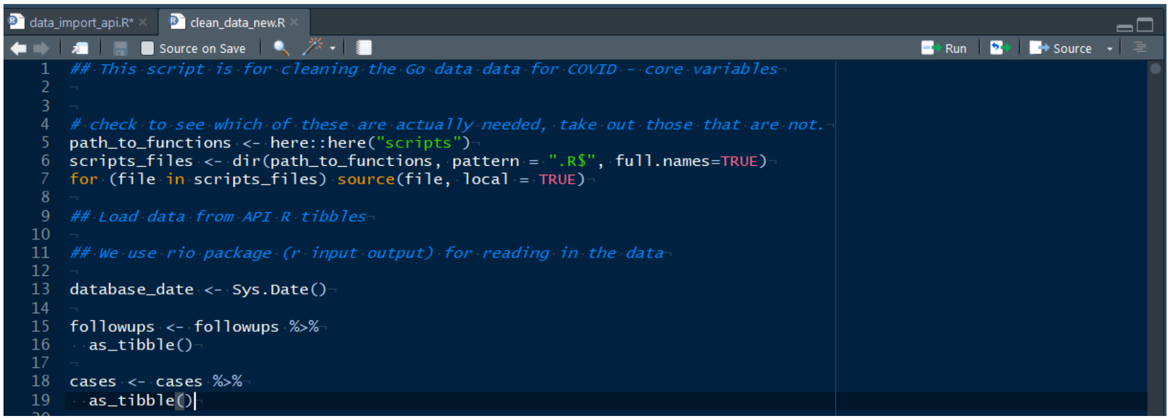
```
Data
cases                  53 obs. of 69 variables
contacts               58 obs. of 28 variables
content                List of 1
followups              490 obs. of 79 variables
locations              213 obs. of 15 variables
relationships          100 obs. of 37 variables
response               List of 10
response_cases         List of 10
response_contacts      List of 10
response_followups     Large response (10 elements, 1.2 Mb)
response_locations     List of 10
response_relationshi…  List of 10
response_teams         List of 10
response_users         List of 10
teams                  8 obs. of 9 variables
users                  133 obs. of 45 variables
Values
```

NOTE: please switch your language to English in your Go.Data instance before running this API script, to ensure core data elements are all brought back in a consistent form

Step 3. Run cleaning script to clean and export dataframes

The dataframes as retrieved straight from API contain many nested arrays in lists. In order to properly un-nest relevant fields and do some basic data manipulation to these data frames before exporting to .CSV (or prepping for additional analysis in R):

Navigate to **02_clean_data_api.R** (also in **report_sources** folder) and run script



```r
## This script is for cleaning the Go data data for COVID - core variables


# check to see which of these are actually needed, take out those that are not.
path_to_functions <- here::here("scripts")
scripts_files <- dir(path_to_functions, pattern = ".R$", full.names=TRUE)
for (file in scripts_files) source(file, local = TRUE)

## Load data from API R tibbles

## We use rio package (r input output) for reading in the data

database_date <- Sys.Date()

followups <- followups %>%
  as_tibble()

cases <- cases %>%
  as_tibble()
```

.

This will result in the following cleaned .csv files saved in the **data > clean > csv** folder, with format matching the pattern below:

- *contacts_clean_[today's date].csv*
- *cases_clean_[today's date].csv*
- *followups_clean_[today's date].csv*
- *relationships_clean_[today's date].csv*
- etc

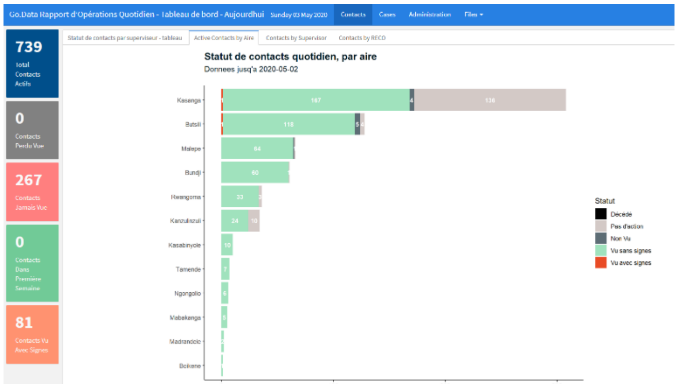
You will also have .rds files in the **data > clean > rds** folder, with no date, updated each time you run script to be most recent. This condensed format will be used for subsequent analysis scripts if desired:

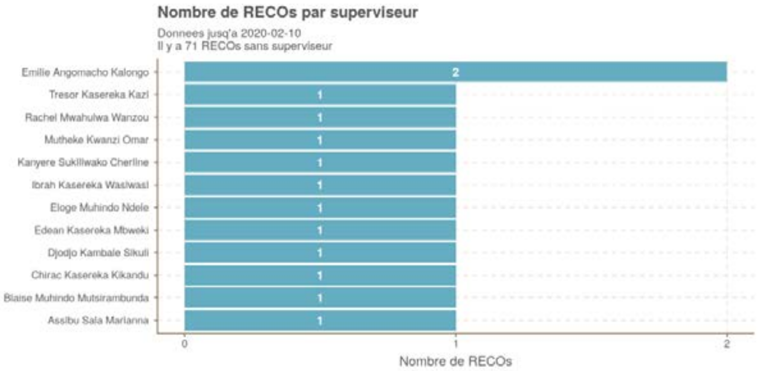
- *contacts_clean.rds*
- etc

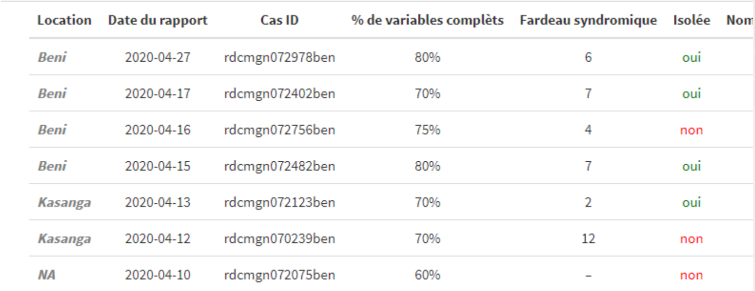
NOTE: these cleaning scripts focus on the CORE data variables and not custom questionnaire variables, as questionnaires are configurable for each country or institution deploying Go.Data. No core data elements (those living outside of questionnaires) should need updating in terms of coding; however, if you would like to pull in additional questionnaire data elements you may need to slightly modify this script to accommodate these extra fields. Additionally, it is possible that your location hierarchy or team structure may vary in your deployment setting (I.e. supervisor registered at a different admin level) so changes may need to be made to the location cleaning scripts. If you encounter any issues here please reach out to your Go.Data focal point and they can assist to help you troubleshoot.

Step 4. Utilize cleaned datasets for additional analysis inside or outside of R

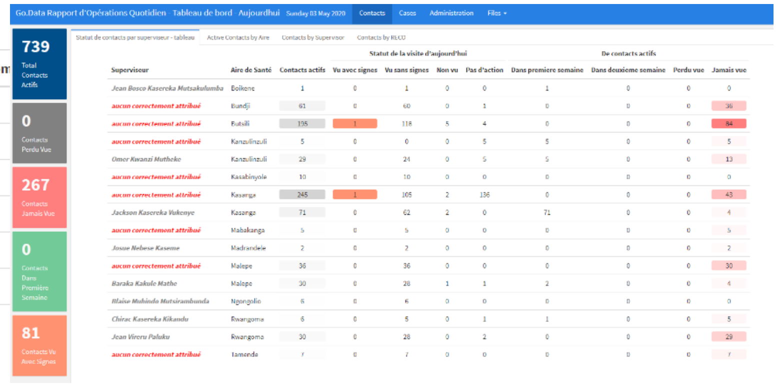
The cleaned datasets will now be much easier to do additional analysis whether inside or outside of R. We have created some sample scripts to get you started in some basic dashboard analyses (see, for example, **03_daily_dashboard_today_current.Rmd** for a ready-made HTML dashboard that will give you stats on a range of operational metrics to be monitored by supervisor and contact tracer on a daily basis). Screenshots below show some of these graphics, such as contact follow-up status by a given admin level and a follow-up table by supervisor. Additionally, you can build on these reports to make more customized analyses.







Location	Date du rapport	Cas ID	% de variables complèts	Fardeau syndromique	Isolée	Nom
Beni	2020-04-27	rdcmgn072978ben	80%	6	oui	
Beni	2020-04-17	rdcmgn072402ben	70%	7	oui	
Beni	2020-04-16	rdcmgn072756ben	75%	4	non	
Beni	2020-04-15	rdcmgn072482ben	80%	7	oui	
Kasanga	2020-04-13	rdcmgn072123ben	70%	2	oui	
Kasanga	2020-04-12	rdcmgn070239ben	70%	12	non	
NA	2020-04-10	rdcmgn072075ben	60%	-	non	


Go.Data Rapport d'Opérations Quotidien - Tableau de bord - Aujourdhui
Sunday 03 May 2020
Contacts
Cases
Administration
Files
739 Total Contacts Actifs
0 Contacts Perdu Vue
267 Contacts Jamais Vue
0 Contacts Dans Première Semaine
81 Contacts Vu Avec Signes
Statut de contacts par superviseur - tableau
Active Contacts by Aire
Contacts by Supervisor
Contacts by RECO

			Statut de la visite d'aujourd'hui				De contacts actifs			
Superviseur	Aire de Santé	Contacts actifs	Vu avec signes	Vu sans signes	Non vu	Pas d'action	Dans premiere semaine	Dans deuxieme semaine	Perdu vue	Jamais vue
Jean Bosco Kasereka Mutsakulumba	Boikene	1	0	1	0	0	1	0	0	0
aucun correctement attribué	Bundji	61	0	60	0	1	0	0	0	36
aucun correctement attribué	Butsili	195	1	118	5	4	0	0	0	84
aucun correctement attribué	Kanzulinzuli	5	0	0	0	5	5	0	0	5
Omer Kwanzi Mutheke	Kanzulinzuli	29	0	24	0	5	5	0	0	13
aucun correctement attribué	Kasabinyole	10	0	10	0	0	0	0	0	0
aucun correctement attribué	Kasanga	245	1	105	2	136	0	0	0	43
Jackson Kasereka Vukenye	Kasanga	71	0	62	2	0	71	0	0	4
aucun correctement attribué	Mabakanga	5	0	5	0	0	0	0	0	5
Josue Mebese Kaseme	Madrandele	2	0	2	0	0	0	0	0	2
aucun correctement attribué	Malepe	36	0	36	0	0	0	0	0	30
Baraka Kakule Mathe	Malepe	30	0	28	1	1	2	0	0	4
Blaise Muhindo Mutsirambunda	Ngongolio	6	0	6	0	0	0	0	0	0
Chirac Kasereka Kikandu	Rwangoma	6	0	5	0	1	1	0	0	5
Jean Vireru Paluku	Rwangoma	30	0	28	0	2	0	0	0	29
aucun correctement attribué	Tamende	7	0	7	0	0	0	0	0	7

Please check back frequently at our Community of Practice site where we will be maintaining active discussions across the user base on different analytics learnings and methodologies that can be applied across settings.