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) – download here: https://www.r-project.org/
- Valid Go.Data login credentials for the instance you are accessing
- 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.
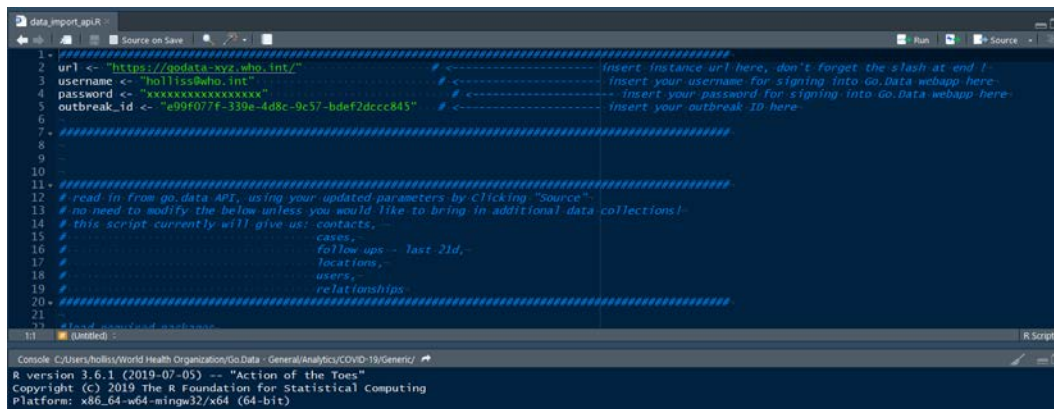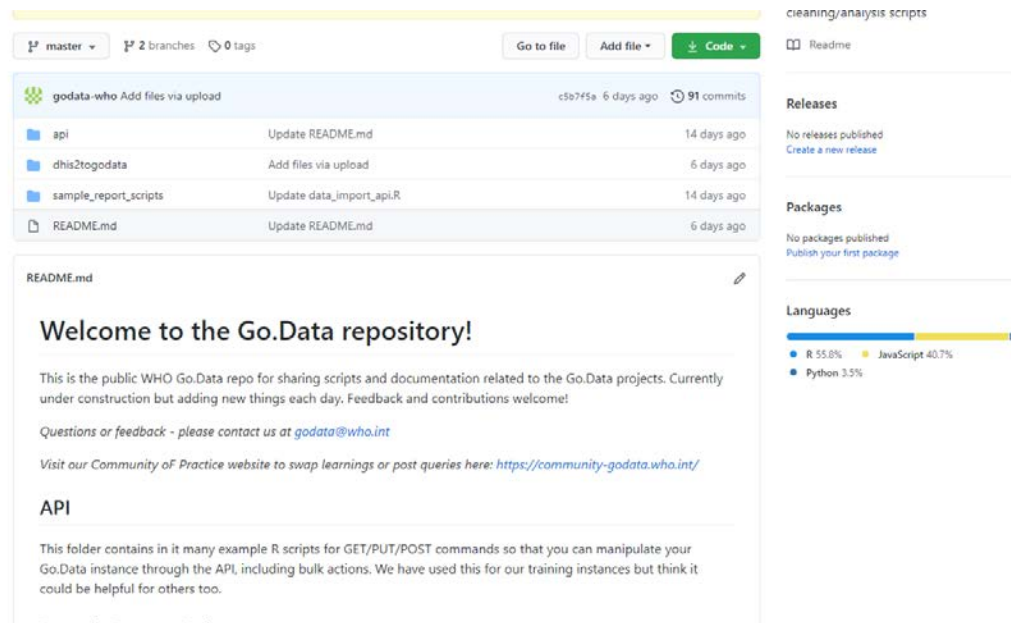

The folder hierarchy contains a **godata.Rproj** file from where you can double-click to start your R session, **report_sources** folder that contains R scripts necessary to execute, **report_outputs** folder where your HTML reports will be saved to, **data** folder where your cleaned data will be saved 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.

Step 2. Run data_import_api 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.
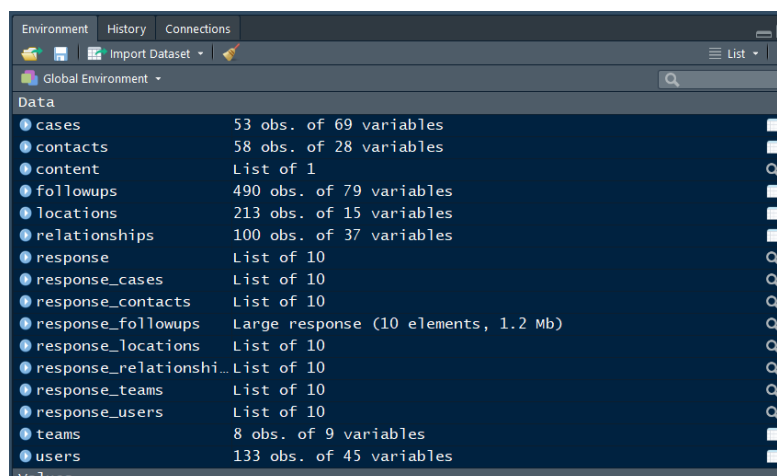

In order to obtain your outbreak ID, navigate to View Outbreak in Go.Data and you can find it in the URL.
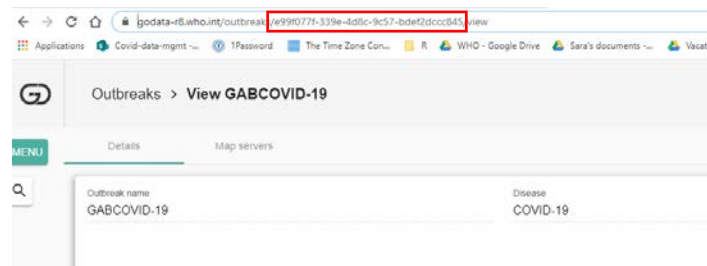
At the end of this script, you should have created the following data frames in your R global environment that will be used in subsequent cleaning scripts: **cases; contacts; followups; locations; outbreaks; relationships; teams; users**

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

This will result in the following cleaned .csv files saved in the **data > clean** folder:

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

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.

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.