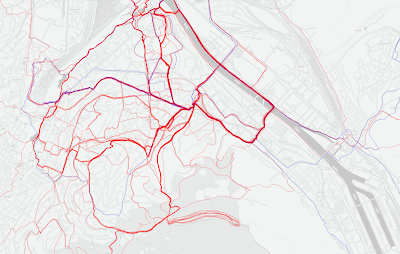
Have fun and leave a comment if you have any questions.

[](https://i2.wp.com/2.bp.blogspot.com/-602gK0AN3HM/Wugy8RJe1oI/AAAAAAAADQc/vd8HvXNqv5sJnX0E0Nb5dRVwsOZecsuTwCLcBGAs/s1600/network.PNG?ssl=1)

options(stringsAsFactors = F)

rm(list=ls())

library(httr)  
library(rjson)  
library(leaflet)  
library(dplyr)

token <- “”

# Functions —————————————————————

get.coord.df.from.stream <- function (stream.obj) {  
  data.frame(lat = sapply(stream.obj[[1]]$data, USE.NAMES = F, FUN = function (x) x[[1]]),  
             lon = sapply(stream.obj[[1]]$data, USE.NAMES = F, FUN = function (x) x[[2]]))  
}

get.stream.from.activity <- function (act.id, token) {  
  stream <- GET(“https://www.strava.com/”,  
                path = paste0(“api/v3/activities/”, act.id, “/streams/latlng”),  
                query = list(access\_token = token))  
  content(stream)  
}

get.activities2 <- function (token) {  
  activities <- GET(“https://www.strava.com/”, path = “api/v3/activities”,  
                    query = list(access\_token = token, per\_page = 200))  
  activities <- content(activities, “text”)  
  activities <- fromJSON(activities)  
  res.df <- data.frame()  
  for (a in activities) {  
    values <- sapply(c(“name”, “distance”, “moving\_time”, “elapsed\_time”, “total\_elevation\_gain”,  
                       “type”, “id”, “start\_date\_local”,  
                       “location\_country”, “average\_speed”, “max\_speed”, “has\_heartrate”, “elev\_high”,  
                       “elev\_low”, “average\_heartrate”, “max\_heartrate”), FUN = function (x) {  
                         if (is.null(a[[x]])) {  
                           NA } else { a[[x]] }  
                       })  
    res.df <- rbind(res.df, values)  
  }  
  names(res.df) <- c(“name”, “distance”, “moving\_time”, “elapsed\_time”, “total\_elevation\_gain”,  
                     “type”, “id”, “start\_date\_local”,  
                     “location\_country”, “average\_speed”, “max\_speed”, “has\_heartrate”, “elev\_high”,  
                     “elev\_low”, “average\_heartrate”, “max\_heartrate”)  
  res.df  
}

get.multiple.streams <- function (act.ids, token) {  
  res.list <- list()  
  for (act.id.i in 1:length(act.ids)) {  
    if (act.id.i %% 5 == 0) cat(“Actitivy no.”, act.id.i, “of”, length(act.ids), “\n”)  
    stream <- get.stream.from.activity(act.ids[act.id.i], token)  
    coord.df <- get.coord.df.from.stream(stream)  
    res.list[[length(res.list) + 1]] <- list(act.id = act.ids[act.id.i],  
                                             coords = coord.df)  
  }  
  res.list  
}

activities <- get.activities2(token)

stream.list <- get.multiple.streams(activities$id, token)  
  
# Leaflet —————————————————————–

lons.range <- c(9.156572, 9.237580)  
lats.range <- c(48.74085, 48.82079)

map <- leaflet() %>%  
  addProviderTiles(“OpenMapSurfer.Grayscale”, # nice: CartoDB.Positron, OpenMapSurfer.Grayscale, CartoDB.DarkMatterNoLabels   
                   options = providerTileOptions(noWrap = T)) %>%  
  fitBounds(lng1 = min(lons.range), lat1 = max(lats.range), lng2 <- max(lons.range), lat2 = min(lats.range))

add.run <- function (act.id, color, act.name, act.dist, strlist = stream.list) {  
  act.ind <- sapply(stream.list, USE.NAMES = F, FUN = function (x) {  
    x$act.id == act.id  
  })  
  act.from.list <- strlist[act.ind][[1]]  
  map <<- addPolylines(map, lng = act.from.list$coords$lon,  
               lat = act.from.list$coords$lat,  
               color = color, opacity = 1/3, weight = 2,  
               popup = paste0(act.name, “, “, round(as.numeric(act.dist) / 1000, 2), ” km”))  
}

# plot all  
for (i in 1:nrow(activities)) {  
  add.run(activities[i, “id”], ifelse(activities[i, “type”] == “Run”, “red”,  
                                      ifelse(activities[i, “type”] == “Ride”, “blue”, “black”)),  
          activities[i, “name”], activities[i, “distance”])  
}