Download Rijkswaterstaat MWTL data from the Data Distribution Layer

Get DDL data

```
library(tidyverse)
library(lubridate)
# devtools::install_github("wstolte/rwsapi")
library(rwsapi)
library(rlist)
library(httr)

# A description (in dutch) of the RWS api is found here:
# https://rahuls01.github.io/?json#introduction
# I wrote a package to make it a bit easier (rwsapi)
# Still not easy. Any suggestions or contributions are very welcome.
# first, download the metadata catalogue from the RWS data service

# read the metadata set
# newMetadata = T
# if(newMetadata){
  metadata <- rwsapi::rws_metadata()
  parmeta <- metadata$content$AquoMetadataLijst %>% rlist::list.flatten()%>% as_tibble() %>% rename_all(tolower)
  locmeta <- metadata$content$LocatieLijst %>% rename_all(tolower)
  mapping <- metadata$content$AquoMetadataLocatieLijst %>% rename_all(tolower)
  ddlMetadata <- parmeta %>% full_join(mapping) %>% full_join(locmeta)
#   write_delim(ddlMetadata, "ddlMetadata.csv", delim = ";")
# } else{
#   ddlMetadata <- read_delim("ddlMetadata.csv", delim = ";")
# }

# find parameter # something with pH = "zuurgraad" in Dutch
# This is in the used AQUO standard a quantity (grootheid)
# nitrate, for example is a parameter, with grootheid == "CONCTTE" (concentration)
# Salinity and temperature are also quantities

grootheid = "zuurgraad"

# find exact parameter description
myPar <- ddlMetadata %>%
  filter(grepl(grootheid, tolower(ddlMetadata$parameter_wat_omschrijving))) %>%
  distinct(parameter_wat_omschrijving) %>% unlist() %>% unname()

myPar # looks good

# testing

location = "NOORDWK20" # 20 km off the coast at Noordwijk
firstYear = 2000
lastYear = 2020

getDDLdata <- function(parameter_wat_omschrijving, location, firstYear, lastYear, metadata){
  myMetadata <- metadata %>%
    filter(parameter_wat_omschrijving == parameter, code == location) %>%
    rename(locatie.code = code)

  beginDatumTijd <- paste0(firstYear, "-01-01T00:00:00.000+01:00")
  eindDatumTijd <- paste0(lastYear, "-12-31T23:59:59.000+01:00")

  getlist <- rwsapi::rws_makeDDLapiList(mijnCatalogus = myMetadata, beginDatumTijd, eindDatumTijd, "OW")

  df <- rwsapi::rws_observations2(getlist[[1]])$content
  return(df)
}

# test
```

```
df <- getDDLdata(myPar, location, firstYear , lastYear, ddlMetadata)
## all pH data for this location

# vind locations where pH is measured (these are also fresh water locations!!!)
#
locs <- ddlMetadata %>%
  filter(parameter_wat_omschrijving == myPar) %>%
  distinct(code, x,y, coordinatenstelsel)

# from here, you would want to do a spatial query and
# get data while looping over getDDLdata
require(sf)
locations <- locs %>%
  st_as_sf(coords = c("x", "y"), crs = 25831) %>%
  st_transform(4326)   #  transform to wgs84

# Marine regions from marineregions.org (VLIZ)
layerurl <- "http://geo.vliz.be/geoserver/MarineRegions/ows?service=WFS&version=1.0.0
&request=GetFeature&typeName=MarineRegions:eez_iho_union_v2&outputFormat=application/json"
marineRegions <- sf::st_read(layerurl)

# play a bit with the buffer (decimial degrees).
# if zero, many freshwater stations are also selected.
marineLocs <- locations %>% sf::st_intersection(st_buffer(marineRegions, -0.05))

marineRegions %>% st_intersection(marineLocs) %>%
  ggplot() +
  geom_sf()

myLocs <- marineLocs %>%
  st_drop_geometry() %>%
  distinct(code) %>%
  unlist() %>% unname()

# then loop function getDDLdata over locs

for(ii in 1:length(myLocs)){
  df <- getDDLdata(myPar, myLocs[ii], firstYear , lastYear, ddlMetadata)
  write_delim(df, paste(myPar,myLocs[ii], ".csv"))
}
```