Package 'QBMS'

March 7, 2024

Type Package

Title Query the Breeding Management System(s)

Version 1.0.0

Date 2024-03-07

Author Khaled Al-Shamaa [aut, cre],
Mariano Omar Crimi [ctb],
Zakaria Kehel [ctb],
Johan Aparicio [ctb],
ICARDA [cph]

Maintainer Khaled Al-Shamaa <k.el-shamaa@cgiar.org>

Description This R package assists breeders in linking data systems with their analytic pipelines, a crucial step in digitizing breeding processes. It supports querying and retrieving phenotypic and genotypic data from systems like 'EBS' <https://ebs.excellenceinbreeding.org/>, 'BMS' <https://bmspro.io>, 'BreedBase' <https://breedbase.org>, and 'GIGWA' <https://github.com/SouthGreenPlatform/Gigwa2> (using 'BrAPI' <https://brapi.org> calls).
Extra helper functions support environmental data sources, including 'TerraClimate' <https://www.climatologylab.org/terraclimate.html> and 'FAO' 'HWSDv2' <https://gaez.fao.org/pages/hwsd> soil database.

License GPL (>= 3)

URL https://icarda-git.github.io/QBMS/

BugReports https://github.com/icarda-git/QBMS/issues

Depends R (>= 3.1.0)

Imports httr, jsonlite, tcltk, utils, RNetCDF, stats, terra, RSQLite, DBI

Suggests knitr, rmarkdown, remotes, async

VignetteBuilder knitr

Encoding UTF-8

NeedsCompilation no

RoxygenNote 7.3.1

Repository CRAN

Date/Publication 2024-03-07 09:40:09 UTC

R topics documented:

`brapi_get_call` *Internal Function Used for Core BrAPI GET Calls*

Description

This function is created for *internal use only* to call BrAPI in the GET method and retrieve the raw response data and send back the results. This function takes care of pagination, authentication, encoding, compression, decoding JSON response, etc.

Usage

```
brapi_get_call(call_url, nested = TRUE)
```

Arguments

`call_url`	BrAPI URL to call in GET method.
`nested`	Logical indicating whether retrieved JSON data will be flattened. Default is TRUE.

Value

Result object returned by the JSON API response.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

brapi_headers *Common HTTP Headers to Send*

Description

Builds the list of common HTTP headers to send with each API call.

Usage

```
brapi_headers()
```

Value

A list of common HTTP headers to send.

build_pedigree_table *Building Pedigree Table Recursively*

Description

Internal helper function to build the pedigree table recursively.

Usage

```
build_pedigree_table(
  geno_list = NULL,
  pedigree_list = NULL,
  pedigree_df = NULL
)
```

Arguments

geno_list	A list of genotypes/germplasm names.
pedigree_list	A list of associated pedigree strings.
pedigree_df	Pedigree data.frame as per the previous call/iteration.

Value

A data.frame with three columns corresponding to the identifiers for the individual, female parent, and male parent, respectively.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

`calc_biovars` *Calculate the Bioclimatic Variables*

Description

Bioclimatic variables are derived from the monthly temperature and rainfall values in order to generate more biologically meaningful variables. These are often used in species distribution modeling and related ecological modeling techniques. The bioclimatic variables represent annual trends (e.g., mean annual temperature, annual precipitation) seasonality (e.g., annual range in temperature and precipitation) and extreme or limiting environmental factors (e.g., temperature of the coldest and warmest month, and precipitation of the wet and dry quarters). A quarter is a period of three months (1/4 of the year).

They are coded as follows:

- BIO1 = Annual Mean Temperature
- BIO2 = Mean Diurnal Range (Mean of monthly (max temp - min temp))
- BIO3 = Isothermality (BIO2/BIO7) (* 100)
- BIO4 = Temperature Seasonality (standard deviation *100)
- BIO5 = Max Temperature of Warmest Month
- BIO6 = Min Temperature of Coldest Month
- BIO7 = Temperature Annual Range (BIO5-BIO6)
- BIO8 = Mean Temperature of Wettest Quarter
- BIO9 = Mean Temperature of Driest Quarter
- BIO10 = Mean Temperature of Warmest Quarter
- BIO11 = Mean Temperature of Coldest Quarter
- BIO12 = Annual Precipitation
- BIO13 = Precipitation of Wettest Month
- BIO14 = Precipitation of Driest Month
- BIO15 = Precipitation Seasonality (Coefficient of Variation)
- BIO16 = Precipitation of Wettest Quarter
- BIO17 = Precipitation of Driest Quarter
- BIO18 = Precipitation of Warmest Quarter
- BIO19 = Precipitation of Coldest Quarter

This work is derivative from the dismo R package

Usage

```
calc_biovars(data)
```

Arguments

data	Data.frame has 4 mandatory columns (year, ppt, tmin, and tmax), and 12 rows (months) for each year sorted from Jan to Dec.

Value

Data.frame has 19 columns for "bioclim" variables (bio1-bio19) and last column for year (you will get one row per year).

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

Robert Hijmans, Museum of Vertebrate Zoology, UC Berkeley

References

Nix, 1986. A biogeographic analysis of Australian elapid snakes. In: R. Longmore (ed.). Atlas of elapid snakes of Australia. Australian Flora and Fauna Series 7. Australian Government Publishing Service, Canberra.

---

debug_qbms *Debug Internal QBMS Status Object*

Description

Returns the internal QBMS status object for debugging purposes.

Usage

```
debug_qbms()
```

Value

An environment object containing package configuration and status.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

Examples

```
obj <- debug_qbms()
obj$config
obj$state
```

get_async_page *Async Version of HTTP GET Request*

Description

A small helper function to create an 'async' version of the original HTTP GET request.

Usage

```
get_async_page(full_url, nested)
```

Arguments

full_url	URL to retrieve.
nested	Logical indicating whether to flatten nested data frames. Default is FALSE.

Value

Async version of the HTTP GET request.

get_async_pages *Run All Supplied Pages*

Description

A small helper function to create a deferred value that is resolved when all listed pages are resolved.

Usage

```
get_async_pages(pages, nested)
```

Arguments

pages	List of URLs to retrieve.
nested	Logical indicating whether to flatten nested data frames. Default is FALSE.

Value

Async deferred object.

get_germplasm_attributes *Retrieve Attributes for a Specified Germplasm*

Description

Retrieve Attributes for a Specified Germplasm

Usage

```
get_germplasm_attributes(germplasm_name = "")
```

Arguments

germplasm_name	The name of the germplasm.

Value

A data frame containing the attributes of the specified germplasm.

Author(s)

Johan Steven Aparicio, <j.aparicio@cgiar.org>

See Also

login_bms, set_crop, get_germplasm_data

Examples

```
if (interactive()) {
  # Configure your server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Log in using your account (interactive mode)
  # You can pass your username and password as parameters (batch mode)
  login_bms()

  # Select a crop
  set_crop("wheat")

  # Select a breeding program by name
  set_program("Wheat International Nurseries")

  # Retrieve attribute data of a specified germplasm in a crop
  germplasm_attributes <- get_germplasm_attributes("Jabal")
}
```

get_germplasm_data *Retrieve Observations Data for a Specified Germplasm.*

Description

Retrieves all available observations data for a given germplasm in the current crop database, regardless of the nested structure of programs/trials.

Usage

```
get_germplasm_data(germplasm_name = "")
```

Arguments

germplasm_name	The name of the germplasm.

Value

A data frame containing the aggregated observations data for the germplasm from all trials.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

login_bms, set_crop, get_germplasm_attributes

Examples

```
if (interactive()) {
  # Configure server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Log in using your account (interactive mode)
  # You can pass your username and password as parameters (batch mode)
  login_bms()

  # Select a crop
  set_crop("wheat")

  # Select a breeding program by name
  set_program("Wheat International Nurseries")

  # Retrieve observations data for a specified germplasm aggregated from all trials
  germplasm_observations <- get_germplasm_data("Jabal")
}
```

get_germplasm_id *Get Germplasm ID*

Description

Retrieves the germplasm ID for the given germplasm name in the current crop.

Usage

```
get_germplasm_id(germplasm_name = "")
```

Arguments

germplasm_name	The name of the germplasm.

Value

A string of the germplasm ID.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

set_crop, get_germplasm_data, get_germplasm_attributes

get_germplasm_list *Get the Germplasm List of the Current Active Study*

Description

Retrieves the germplasm list of the current active study as configured in the internal state object using ‘set_study()‘ function.

Usage

```
get_germplasm_list()
```

Value

A data frame of the study germplasm list.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

login_bms, set_crop, set_program, set_trial, set_study

Examples

```
if(interactive()) {
  # Configure your server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Login using your account (interactive mode)
  # You can pass your username and password as parameters (batch mode)
  login_bms()

  # Select a crop by name
  set_crop("wheat")

  # Select a breeding program by name
  set_program("Wheat International Nurseries")

  # Select a specific study/trial by name
  set_trial("IDYT39")

  # Select a specific environment/location dataset
  set_study("IDYT39 Environment Number 9")

  # Retrieve the germplasm list of the selected environment/location
  germplasm <- get_germplasm_list()
}
```

---

get_hwsd2 *Get HWSD v2 Soil Data for a Given Location(s)*

Description

The HWSD2_SMU table contains general information for each of the soil units occurring in any given SMU code (dominant soil unit and up to 11 associated soils).

The SEQUENCE column refers to the sequence in which soil units within the SMU are presented (in order of percentage share). The dominant soil has sequence 1. The sequence can range between 1 and 12.

The SHARE column refers to the share of the soil unit within the mapping unit in percentage. Shares of soil units within a mapping unit always sum up to 100 percent.

The HWSD2_LAYERS table provides soil attributes per depth layer for each of the seven depth layers (D1 to D7) separately (represented in the LAYER column in the HWSD2_LAYERS table). The depth of the top and bottom of each layer is defined in the TOPDEP and BOTDEP columns, respectively.

Usage

```
get_hwsd2(df, con, x = "longitude", y = "latitude", sequence = 1, layer = "D1")
```

Arguments

df	data.frame that list locations info including the coordinates in decimal degree format.
con	the HWSDv2 object returns from the ini_hwsd2() function.
x	longitude column name in the df data.frame (default is 'Longitude').
y	latitude column name in the df data.frame (default is 'Latitude').
sequence	the sequence in which soil units are presented (in order of percentage share). The dominant soil has sequence 1. The sequence can range between 1 and 12.
layer	the depth layer range from 'D1' to 'D7'. The depth of the top and bottom of each layer is defined in the TOPDEP and BOTDEP columns, respectively.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

ini_hwsd2

Examples

```
if (interactive()) {
  # create a simple data.frame for a list of locations and their coordinates
  Location  <- c('Tel-Hadya', 'Terbol', 'Marchouch')
  Latitude  <- c(36.016, 33.808, 33.616)
  Longitude <- c(36.943, 35.991, -6.716)

  sites <- data.frame(Location, Latitude, Longitude)

  # initiate, download, and setup the HWSD v2 in a given local directory
  hwsd2 <- ini_hwsd2(data_path = 'C:/Users/Kel-shamaa/Downloads/HWSD v2/')

  # query soil attributes for given sites using the HWSD v2 connection object
  #
  # sequence parameter, range between 1 and 12 (max), 1 is the dominant soil.
  # returned df has SHARE column refers to share%
  #
  # layer parameter refers to depth layer (D1 to D7).
  # returned df has TOPDEP/BOTDEP columns represent top/bottom layer depth in cm.
  sites <- get_hwsd2(df = sites,
                     con = hwsd2,
                     x = 'Longitude',
                     y = 'Latitude',
                     sequence = 1,
                     layer = 'D1')
}
```

get_login_details *Login Pop-Up Window*

Description

Builds a GUI pop-up window using Tcl/Tk to insert the username and password.

Usage

```
get_login_details()
```

Value

A vector of inserted username and password.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

get_parents *Get Direct Parents*

Description

Internal helper function to split the given pedigree string that provides the parentage through which a cultivar was obtained and retrieve the pedigrees of the direct parents.

Usage

```
get_parents(pedigree)
```

Arguments

pedigree	A string providing the parentage through which a cultivar was obtained.

Value

A vector of two items, representing the direct female and male parents.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

`get_pedigree_table` *Get the Pedigree Table*

Description

Retrieve the pedigree table starting from the current germplasm list and associated pedigree string that provides the parentage through which a cultivar was obtained.

Usage

```
get_pedigree_table(
  data,
  geno_column = "germplasmName",
  pedigree_column = "pedigree"
)
```

Arguments

`data`	Germplasm dataset as a data.frame.
`geno_column`	Name of the column that identifies the genotype/germplasm names.
`pedigree_column`	Name of the column that identifies the pedigree strings.

Value

A data.frame with three columns corresponding to the identifiers for the individual, female parent, and male parent, respectively. The row giving the pedigree of an individual appears before any row where that individual appears as a parent. Founders use NA in the parental columns.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

Examples

```
if (interactive()) {
  # Configure your server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Login using your account (interactive mode)
  # You can pass your username and password as parameters (batch mode)
  login_bms()

  # Select a crop by name
  set_crop("wheat")

  # Select a breeding program by name
  set_program("Wheat International Nurseries")
```

```
  # Select a specific study/trial by name
  set_trial("IDYT39")

  # Select a specific environment/location dataset
  set_study("IDYT39 Environment Number 9")

  # Retrieve the germplasm list of the selected environment/location
  germplasm <- get_germplasm_list()

  pedigree_table <- get_pedigree_table(germplasm, "germplasmName", "pedigree")

  ############################
  # nadiv package way
  # library(nadiv)

  # Get additive relationship matrix in sparse matrix format
  # A <- nadiv::makeA(pedigree_table)

  # Get A inverse matrix using base R function
  # AINV <- solve(as.matrix(A))

  ############################
  # ASReml-R package way
  # library(asreml)

  # Represent A inverse matrix in an efficient way using i, j index and Ainverse value
  # Actual genotype names of any given index are in the attr(ainv, "rowNames")
  # ainv <- asreml::ainverse(pedigree_table)

  ############################
  # Dummy data set for testing
  test <- data.frame(genotype = c("X", "Y"),
                     pedigree = c("A//B/D/2/C", "B/C/3/A//B/C/2/D"))

  pedigree_table <- get_pedigree_table(test, "genotype", "pedigree")
}
```

`get_program_studies` *Get the List of Trials/Studies/Locations Information of the Current Selected Program*

Description

Retrieves all environments/locations information of the trials studies in the current active program as configured in the internal state object using the ‘set_program()‘ function.

Usage

```
get_program_studies()
```

Value

A data frame of locations information for each study in the program trials.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

login_bms, set_crop, set_program

Examples

```
if(interactive()) {
  # config your server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # login using your account (interactive mode)
  # you can pass your username and password as parameters (batch mode)
  login_bms()

  # Select a crop by name
  set_crop("wheat")

  # select a breeding program by name
  set_program("Wheat International Nurseries")

 # retrieve all environments/locations information in the selected program studies/trials
  program_studies <- get_program_studies()
}
```

---

get_program_trials *Internal Function Used to Retrieve the Rough List of Trials*

Description

This function is created for *internal use only* to retrieve the raw list of trials from the pre-selected (i.e., currently active) crop and breeding program combination as already configured in the internal state object using ‘set_crop()‘ and ‘set_program()‘ functions, respectively.

Usage

```
get_program_trials()
```

Value

A list of trials information.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

login_bms, set_crop, set_program, list_trials

---

get_qbms_connection *Get the QBMS Connection*

Description

Retrieves the QBMS connection object from the current environment.

Usage

```
get_qbms_connection()
```

Value

A list containing the current connection configuration and status.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

set_qbms_connection

Examples

```
if(interactive()) {
  # Configure your server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Login using your account (interactive mode)
  # You can pass your username and password as parameters (batch mode)
  login_bms()

  # Select a crop by name
  set_crop("wheat")

  # Select a breeding program by name
  set_program("Wheat International Nurseries")

  # Get germplasm data
  df1 <- get_germplasm_data("Jabal")
```

```
  # Save the current connection (phenotypic server)
  con1 <- get_qbms_connection()

  # Configure QBMS to connect to the genotypic server
  set_qbms_config("https://gigwa.southgreen.fr/gigwa/", engine = "gigwa", no_auth = TRUE)

  # Set the db, project, and run
  gigwa_set_db("DIVRICE_NB")
  gigwa_set_project("refNB")
  gigwa_set_run("03052022")

  # Get associated metadata
  df2 <- gigwa_get_metadata()

  # Save the current connection (before switch)
  con2 <- get_qbms_connection()

  # Load the saved phenotypic server connection
  set_qbms_connection(con1)

  # Continue retrieving germplasm attributes from the phenotypic server
  df3 <- get_germplasm_attributes("Jabal")
}
```

`get_study_data` *Get the Observations Data of the Current Active Study*

Description

Retrieves the observations data of the current active study as configured in the internal state object using ‘set_study()‘ function.

Usage

```
get_study_data()
```

Value

A data frame of the study observations data.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

`login_bms`, `set_crop`, `set_program`, `set_trial`, `set_study`

Examples

```
if(interactive()) {
  # Configure your server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Login using your account (interactive mode)
  # You can pass your username and password as parameters (batch mode)
  login_bms()

  # Select a crop by name
  set_crop("wheat")

  # Select a breeding program by name
  set_program("Wheat International Nurseries")

  # Select a specific study/trial by name
  set_trial("IDYT39")

  # Select a specific environment/location dataset
  set_study("IDYT39 Environment Number 9")

  # Retrieve the data of the selected environment/location
  data <- get_study_data()
}
```

---

get_study_info *Get the Details/Metadata of the Current Active Study*

Description

Retrieves the details/metadata of the current active study as configured in the internal state object using ‘set_study()‘ function.

Usage

```
get_study_info()
```

Value

A data frame of the study details/metadata.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

login_bms, set_crop, set_program, set_trial, set_study

Examples

```
if(interactive()) {
  # Configure your server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Login using your account (interactive mode)
  # You can pass your username and password as parameters (batch mode)
  login_bms()

  # Select a crop by name
  set_crop("wheat")

  # Select a breeding program by name
  set_program("Wheat International Nurseries")

  # Select a specific study/trial by name
  set_trial("IDYT39")

  # Select a specific environment/location dataset
  set_study("IDYT39 Environment Number 9")

  # Retrieve the general information of the selected environment/location
  info <- get_study_info()
}
```

`get_terraclimate` *Get TerraClimate Data for a Given Coordinate(s)*

Description

TerraClimate is a monthly climate dataset for global terrestrial surfaces from 1958-2021. This function enables you to extract climate variables from the hosting server provided by the Idaho University for a given coordinate(s) without a need to download the whole raster files in the netCDF format (~100MB per variable for each year) and provide them in a standard data frame format ready to use in your code. It also calculates the bioclimatic variables using the `calc_biovars` function derivative from the dismo R package.

TerraClimate vs. WorldClim

- 1958-2021 vs. 1970-2000
- 14 vs. 7 climate variables
- ~4 km vs. ~1 km spatial resolution
- need to calculate vs. pre-calculated 19 bioclimatic variables

Usage

```
get_terraclimate(
  lat,
  lon,
  from = "1958-01-01",
  to = "2022-12-31",
  clim_vars = NULL,
  month_mask = NULL,
  offline = FALSE,
  data_path = "./data/"
)
```

Arguments

`lat`	Vector of Latitude(s) in decimal degree format.
`lon`	Vector of Longitude(s) in decimal degree format.
`from`	Start date as a string in the 'YYYY-MM-DD' format.
`to`	End date as a string in the 'YYYY-MM-DD' format.
`clim_vars`	List of all climate variables to be imported. Valid list includes: *aet, def, pet, ppt, q, soil, srad, swe, tmax, tmin, vap, ws, vpd, and PDSI*. Default is NULL for all.
`month_mask`	A list of all months of interest (e.g., planting season: `c(10:12,1:5)`). Default is NULL for all.
`offline`	Extract TerraClimate data from downloaded netCDF files (default is FALSE)
`data_path`	String contains the directory path where downloaded netCDF files exists (default is './data/')

Value

A list of two data.frame(s) for each pair of coordinates:

- **climate:** includes the climate variables (reference).
- **biovars:** includes the calculated bioclimatic variables (reference).

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

References

Abatzoglou, J., Dobrowski, S., Parks, S. *et al.* TerraClimate, a high-resolution global dataset of monthly climate and climatic water balance from 1958-2015. *Sci Data* **5**, 170191 (2018). doi:10.1038/sdata.2017.191

See Also

`ini_terraclimate`

Examples

```
if (interactive()) {
  # data <- get_terraclimate(36.016, 36.943,
  #                          '1979-09-01', '2012-06-30',
  #                          c('ppt', 'tmin', 'tmax'), c(10:12,1:5))
  data <- get_terraclimate(36.016, 36.943, '1979-09-01', '2012-06-30')

  View(data$climate[[1]])

  View(data$biovars[[1]])
}
```

`get_trial_data` *Get the Observations Data of the Current Active Trial*

Description

Retrieves the observations data of the current active trial (i.e., including all studies within) as configured in the internal state object using ‘set_trial()‘ function.

Usage

```
get_trial_data()
```

Value

A data frame of the trial observations data.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

login_bms, set_crop, set_program, set_trial

Examples

```
if(interactive()) {
  # Configure your server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Login using your account (interactive mode)
  # You can pass your username and password as parameters (batch mode)
  login_bms()

  # Select a crop by name
  set_crop("wheat")
```

```
  # Select a breeding program by name
  set_program("Wheat International Nurseries")

  # Select a specific study/trial by name
  set_trial("IDYT39")

  # Retrieve multi-environment trial data
  MET <- get_trial_data()
}
```

`get_trial_obs_ontology` *Get the Traits Ontology/Metadata of the Current Active Trial*

Description

Retrieves the traits ontology/metadata of the current active trial as configured in the internal state object using the 'set_trial()' function.

Usage

```
get_trial_obs_ontology()
```

Value

A data frame of the traits ontology/metadata.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

login_bms, set_crop, set_program, set_trial

Examples

```
if(interactive()) {
  # Configure your server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Login using your account (interactive mode)
  # You can pass your username and password as parameters (batch mode)
  login_bms()

  # Select a crop by name
  set_crop("wheat")
```

```
  # Select a breeding program by name
  set_program("Wheat International Nurseries")

  # Select a specific study/trial by name
  set_trial("IDYT39")

  # Get observation variable ontology
  ontology <- get_trial_obs_ontology()
}
```

---

gigwa_get_metadata *Get the Metadata of the Current Active GIGWA Run*

Description

This function retrieves the metadata of the current active run as configured in the internal state object using the 'gigwa_set_run()' function.

Usage

```
gigwa_get_metadata()
```

Value

A data.frame of all metadata associated with the samples in the selected run.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

set_qbms_config, gigwa_set_run

Examples

```
if (interactive()) {
  # Configure your GIGWA connection
  set_qbms_config("https://gigwa.southgreen.fr/gigwa/",
                  time_out = 300, engine = "gigwa", no_auth = TRUE)

  # Select a database by name
  gigwa_set_db("3kG_10M")

  # Select a project by name
  gigwa_set_project("3003_ind")

  # Select a specific run by name
  gigwa_set_run("1")
```

```
  # Get a list of all samples in the selected run
  metadata <- gigwa_get_metadata()
}
```

---

gigwa_get_samples *Get the Samples List of the Current Active GIGWA Run*

Description

This function retrieves the samples list of the current active run as configured in the internal state object using the ‘gigwa_set_run()‘ function.

Usage

```
gigwa_get_samples()
```

Value

A vector of all samples in the selected run.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

set_qbms_config, gigwa_set_run

Examples

```
if (interactive()) {
  # Configure your GIGWA connection
  set_qbms_config("https://gigwa.southgreen.fr/gigwa/",
                  time_out = 300, engine = "gigwa", no_auth = TRUE)

  # Select a database by name
  gigwa_set_db("Sorghum-JGI_v1")

  # Select a project by name
  gigwa_set_project("Nelson_et_al_2011")

  # Select a specific run by name
  gigwa_set_run("run1")

  # Get a list of all samples in the selected run
  samples <- gigwa_get_samples()
}
```

gigwa_get_variants *Get Available Variants in the Selected GIGWA Run*

Description

Query the variants (e.g., SNPs markers) in the selected GIGWA run that match a given criteria.

Usage

```
gigwa_get_variants(
  max_missing = 1,
  min_maf = 0,
  samples = NULL,
  start = NULL,
  end = NULL,
  referenceName = NULL
)
```

Arguments

max_missing	Maximum missing ratio (by sample) between 0 and 1 (default is 1 for 100%).
min_maf	Minimum Minor Allele Frequency (MAF) between 0 and 1 (default is 0 for 0%).
samples	A list of samples subset (default is NULL, which will retrieve for all samples).
start	Start position of region (zero-based, inclusive) (e.g., 19750802).
end	End position of region (zero-based, exclusive) (e.g., 19850125).
referenceName	Reference sequence name (e.g., '6H' in the Barley LI-AM).

Value

A data.frame that has the first 4 columns describing attributes of the SNP (rs#: variant name, alleles: reference allele / alternative allele, chrom: chromosome name, and pos: position in bp), while the following columns describe the SNP value for a single sample line using numerical coding 0, 1, and 2 for reference, heterozygous, and alternative/minor alleles.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

Examples

```
if (interactive()) {
  # Configure your GIGWA connection
  set_qbms_config("https://gigwa.southgreen.fr/gigwa/",
                  time_out = 300, engine = "gigwa", no_auth = TRUE)

  # Select a database by name
  gigwa_set_db("Sorghum-JGI_v1")
```

```
  # Select a project by name
  gigwa_set_project("Nelson_et_al_2011")

  # Select a specific run by name
  gigwa_set_run("run1")

  marker_matrix <- gigwa_get_variants(max_missing = 0.2,
                                      min_maf = 0.35,
                                      samples = c("ind1", "ind3", "ind7"))
}
```

---

gigwa_list_dbs *List GIGWA Databases*

Description

Get the list of existing databases in the current GIGWA server.

Usage

```
gigwa_list_dbs()
```

Value

A list of existing databases.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

set_qbms_config

Examples

```
if (interactive()) {
  # Configure your GIGWA connection
  set_qbms_config("https://gigwa.southgreen.fr/gigwa/",
                  time_out = 300, engine = "gigwa", no_auth = TRUE)

  # List existing databases in the GIGWA server
  gigwa_list_dbs()
}
```

gigwa_list_projects *Get the List of All Projects in the Selected GIGWA Database*

Description

Retrieves the projects list from the currently active database as configured in the internal configuration object using the ‘gigwa_set_db()‘ function.

Usage

```
gigwa_list_projects()
```

Value

A list of project names.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

set_qbms_config, gigwa_set_db

Examples

```
if (interactive()) {
  # Configure your GIGWA connection
  set_qbms_config("https://gigwa.southgreen.fr/gigwa/",
                  time_out = 300, engine = "gigwa", no_auth = TRUE)

  # Select a database by name
  gigwa_set_db("Sorghum-JGI_v1")

  # List existing projects
  gigwa_list_projects()
}
```

gigwa_list_runs *Get the List of the Run Names Available in the Selected GIGWA Project*

Description

This function retrieves the runs list from the currently active project as configured in the internal configuration object using the ‘gigwa_set_project()‘ function.

Usage

```
gigwa_list_runs()
```

Value

A list of run names.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

set_qbms_config, gigwa_set_project

Examples

```
if (interactive()) {
  # Configure your GIGWA connection
  set_qbms_config("https://gigwa.southgreen.fr/gigwa/",
                  time_out = 300, engine = "gigwa", no_auth = TRUE)

  # Select a database by name
  gigwa_set_db("Sorghum-JGI_v1")

  # Select a project by name
  gigwa_set_project("Nelson_et_al_2011")

  # List all runs in the selected project
  gigwa_list_runs()
}
```

---

gigwa_set_db *Set the Current Active GIGWA Database by Name*

Description

This function updates the current active database in the internal configuration object (including the brapi connection object).

Usage

```
gigwa_set_db(db_name)
```

Arguments

db_name	The name of the database.

Value

No return value.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

set_qbms_config, gigwa_list_dbs

Examples

```
if (interactive()) {
  # Configure your GIGWA connection
  set_qbms_config("https://gigwa.southgreen.fr/gigwa/",
                  time_out = 300, engine = "gigwa", no_auth = TRUE)

  # Select a database by name
  gigwa_set_db("Sorghum-JGI_v1")
}
```

---

gigwa_set_project *Set the Current Active GIGWA Project*

Description

This function updates the current active project in the internal state object using the programDbId retrieved from GIGWA which is associated with the given ‘project_name‘ parameter.

Usage

```
gigwa_set_project(project_name)
```

Arguments

project_name	The name of the project.

Value

No return value.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

set_qbms_config, gigwa_set_db, gigwa_list_projects

Examples

```
if (interactive()) {
  # Configure your GIGWA connection
  set_qbms_config("https://gigwa.southgreen.fr/gigwa/",
                  time_out = 300, engine = "gigwa", no_auth = TRUE)

  # Select a database by name
  gigwa_set_db("Sorghum-JGI_v1")

  # Select a project by name
  gigwa_set_project("Nelson_et_al_2011")
}
```

`gigwa_set_run` *Set the Current Active GIGWA Run*

Description

This function updates the current active run in the internal state object using the 'studyDbIds' retrieved from GIGWA, which are associated with the given 'run_name' parameter.

Usage

```
gigwa_set_run(run_name)
```

Arguments

`run_name`	The name of the run.

Value

No return value.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

`set_qbms_config`, `gigwa_set_project`, `gigwa_list_runs`

Examples

```
if (interactive()) {
  # Configure your GIGWA connection
  set_qbms_config("https://gigwa.southgreen.fr/gigwa/",
                  time_out = 300, engine = "gigwa", no_auth = TRUE)

  # Select a database by name
```

```
  gigwa_set_db("Sorghum-JGI_v1")

  # Select a project by name
  gigwa_set_project("Nelson_et_al_2011")

  # Select a specific run by name
  gigwa_set_run("run1")
}
```

`ini_hwsd2` *Download and Setup HWSD v2.0 Data Files to Extract their Data Offline*

Description

Download and Setup HWSD v2.0 Data Files to Extract their Data Offline

Usage

```
ini_hwsd2(data_path = "./data/", timeout = 300)
```

Arguments

`data_path`	String containing the directory path where downloaded HWSD v2.0 data files exist (default is './data/').
`timeout`	Timeout in seconds to download each HWSD v2.0 data file (default is 300).

Value

HWSDv2 object that has a list of two items: the HWSD2 raster and the HWSD2 SQLite connection.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

`get_hwsd2`

`ini_terraclimate` *Download TerraClimate netCDF Data Files to Extract their Data Offline*

Description

Download TerraClimate netCDF Data Files to Extract their Data Offline

Usage

```
ini_terraclimate(
  from = "2019-09-01",
  to = "2022-06-30",
  clim_vars = c("ppt", "tmin", "tmax"),
  data_path = "./data/",
  timeout = 300
)
```

Arguments

`from`	Start date as a string in the 'YYYY-MM-DD' format.
`to`	End date as a string in the 'YYYY-MM-DD' format.
`clim_vars`	List of all climate variables to be imported. Valid list includes: *aet, def, pet, ppt, q, soil, srad, swe, tmax, tmin, vap, ws, vpd, and PDSI*. Default is NULL for all.
`data_path`	String containing the directory path where downloaded netCDF files exist (default is './data/')
`timeout`	Timeout in seconds to download each netCDF raster file (default is 300).

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

`get_terraclimate`

Examples

```
if (interactive()) {
  ini_terraclimate('2018-09-01', '2019-06-30', c('ppt', 'tmin', 'tmax'))

  x <- c(-6.716, 35.917, 76.884)
  y <- c(33.616, 33.833, 23.111)

  a <- get_terraclimate(y, x, '2018-09-01', '2019-06-30', c('ppt', 'tmin', 'tmax'))

  a$climate[[1]]
```

```
  a$biovars[[1]]

 b <- get_terraclimate(y, x, '2018-09-01', '2019-06-30', c('ppt', 'tmin', 'tmax'), offline = TRUE)

  b$climate[[1]]
  b$biovars[[1]]
}
```

`list_crops` *Get the List of Supported Crops*

Description

Retrieves the list of supported crops.

Usage

```
list_crops()
```

Value

A list of supported crops.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

login_bms

Examples

```
if(interactive()) {
  # Configure your BMS connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Login using your BMS account (interactive mode)
  # You can pass the BMS username and password as parameters (batch mode)
  login_bms()

  # List supported crops in the BMS server
  list_crops()
}
```

list_locations *Get the List of Locations Information of the Current Selected Crop*

Description

Retrieves the locations information of the current active crop as configured in the internal state object using the 'set_crop()' function.

Usage

```
list_locations()
```

Value

A data frame of the locations information.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

login_bms, set_crop

list_programs *Get the List of Breeding Programs Names*

Description

Retrieves the breeding programs list from the current active crop as configured in the internal configuration object using 'set_crop()' function.

Usage

```
list_programs()
```

Value

A list of breeding programs names.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

login_bms, set_crop

Examples

```
if(interactive()) {
  # Configure your server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Login using your account (interactive mode)
  # You can pass your username and password as parameters (batch mode)
  login_bms()

  # Select a crop by name
  set_crop("wheat")

  # List existing breeding programs
  list_programs()
}
```

`list_studies` *Get the List of Studies in the Current Active Trial*

Description

Retrieves the studies list from the current active trial as configured in the internal state object using ‘set_trial()‘ function.

Usage

```
list_studies()
```

Value

A list of study and location names.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

login_bms, set_crop, set_program, set_trial

Examples

```
if(interactive()) {
  # Configure your server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Login using your account (interactive mode)
  # You can pass your username and password as parameters (batch mode)
```

```
  login_bms()

  # Select a crop by name
  set_crop("wheat")

  # Select a breeding program by name
  set_program("Wheat International Nurseries")

  # Select a specific study/trial by name
  set_trial("IDYT39")

  # List all environments/locations information in the selected study/trial
  list_studies()
}
```

`list_trials` *Get the List of Trials in the Current Active Breeding Program*

Description

Retrieves the trials list from the current active breeding program as configured in the internal state object using 'set_program()' function.

Usage

```
list_trials(year = NULL)
```

Arguments

`year`	The starting year to filter the list of trials (optional, default is NULL).

Value

A list of trials names.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

`login_bms`, `set_crop`, `set_program`

Examples

```
if(interactive()) {
  # Configure your server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Login using your account (interactive mode)
  # You can pass your username and password as parameters (batch mode)
  login_bms()

  # Select a crop by name
  set_crop("wheat")

  # Select a breeding program by name
  set_program("Wheat International Nurseries")

  # List all studies/trials in the selected program
  list_trials()

  # Filter listed studies/trials by year
  list_trials(2022)
}
```

---

login_bms *Login to the Server*

Description

Connects to the server. If the username or password parameters are missing, then a login window will pop up to insert the username and password.

All other connection parameters (i.e., server IP or domain, connection port, API path, and connection protocol e.g., http://) will be retrieved from the qbms_config list.

This function will update both the qbms_config list (brapi connection object in the con key) and the qbms_state list (token value in the token key).

Usage

```
login_bms(username = NULL, password = NULL, encoding = "json")
```

Arguments

username	The username (optional, default is NULL).
password	The password (optional, default is NULL).
encoding	How should the named list body be encoded? Can be one of form (application/x-www-form-urlencoded), multipart (multipart/form-data), or json (application/json).

Value

No return value.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

Examples

```
if(interactive()) {
  # Configure your BMS connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Login using your BMS account (interactive mode)
  # You can pass the BMS username and password as parameters (batch mode)
  login_bms()
}
```

---

login_breedbase *Login to the BreedBase Server*

Description

Logs in to the BreedBase server.

Usage

```
login_breedbase(username = NULL, password = NULL)
```

Arguments

username	The username (optional, default is NULL).
password	The password (optional, default is NULL).

Value

No return value.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

`login_gigwa` *Login to the GIGWA Server*

Description

Connect to the GIGWA server. If the username or password parameters are missing, a login window will pop up to insert the username and password.

All other connection parameters (i.e., server IP or domain, connection port, API path, and connection protocol e.g., http://) will be retrieved from the 'qbms_config' list.

This function will update both the qbms_config list (brapi connection object in the con key) and qbms_state list (token value in the token key).

Usage

```
login_gigwa(username = NULL, password = NULL)
```

Arguments

`username`	The GIGWA username (optional, default is NULL).
`password`	The GIGWA password (optional, default is NULL).

Value

No return value.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

Examples

```
if (interactive()) {
  # Configure your GIGWA connection
 set_qbms_config("http://localhost:59395/gigwa/index.jsp", time_out = 300, engine = "gigwa")

  # Login using your GIGWA account (interactive mode)
  login_gigwa()

  # You can pass GIGWA username and password as parameters (batch mode)
  # login_gigwa("gigwadmin", "nimda")
}
```

`login_oauth2` *Login Using OAuth 2.0 Authentication*

Description

If the request for an access token is valid, the authorization server needs to generate an access token and return these to the client, typically along with some additional properties about the authorization.

Usage

```
login_oauth2(
  authorize_url,
  access_url,
  client_id,
  client_secret = NULL,
  redirect_uri = "http://localhost:1410",
  oauth2_cache = FALSE
)
```

Arguments

`authorize_url`	URL to send the client for authorization.
`access_url`	URL used to exchange unauthenticated for authenticated token.
`client_id`	Consumer key, also sometimes called the client ID.
`client_secret`	Consumer secret, also sometimes called the client secret.
`redirect_uri`	The URL that the user will be redirected to after authorization is complete (default is http://localhost:1410).
`oauth2_cache`	A logical value or a string. TRUE means to cache using the default cache file .httr-oauth, FALSE means don't cache, and NA means to guess using some sensible heuristics. A string means use the specified path as the cache file. Default is FALSE (i.e., don't cache).

Value

No return value.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

rbindlistx *Make One Data.Table from a List of Many*

Description

Performs the equivalent of do.call("rbind", x) on data.frames, but much faster.

Usage

```
rbindlistx(x)
```

Arguments

x	A list containing data.table, data.frame, or list objects.

Value

An unkeyed data.table containing a concatenation of all the items passed in.

rbindx *Combine Data Frames by Row, Filling in Missing Columns*

Description

Combines a list of data frames by row, filling in missing columns with NA.

Usage

```
rbindx(..., dfs = list(...))
```

Arguments

...	The first argument data frame.
dfs	Input data frames to row bind together.

Value

A single data frame.

scan_brapi_endpoints *Scan BrAPI Endpoints*

Description

Scan available BrAPI endpoints on the configured source server.

Usage

```
scan_brapi_endpoints(programDbId = 0, trialDbId = 0, studyDbId = 0)
```

Arguments

programDbId	(numeric) ProgramDbId used for BrAPI endpoints scanning. Default is 0.
trialDbId	(numeric) TrialDbId used for BrAPI endpoints scanning. Default is 0.
studyDbId	(numeric) StudyDbId used for BrAPI endpoints scanning. Default is 0.

Value

A data frame listing the QBMS function, BrAPI endpoint URL, and status.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

set_crop *Set the Current Active Crop*

Description

Updates the current active crop in the internal configuration object (including the BrAPI connection object).

Usage

```
set_crop(crop_name)
```

Arguments

crop_name	The name of the crop.

Value

No return value.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

login_bms, list_crops

Examples

```
if(interactive()) {
  # Configure your server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Login using your account (interactive mode)
  # You can pass your username and password as parameters (batch mode)
  login_bms()

  # Select a crop by name
  set_crop("wheat")
}
```

---

set_program *Set the Current Active Breeding Program*

Description

Updates the current active breeding program in the internal state object using the programDbId which is associated with the given program_name parameter.

Usage

```
set_program(program_name)
```

Arguments

program_name	The name of the breeding program.

Value

No return value.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

login_bms, set_crop, list_programs

Examples

```
if(interactive()) {
  # Configure your server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Login using your account (interactive mode)
  # You can pass your username and password as parameters (batch mode)
  login_bms()

  # Select a crop by name
  set_crop("wheat")

  # Select a breeding program by name
  set_program("Wheat International Nurseries")
}
```

---

set_qbms_config *Configure BMS Server Settings*

Description

Sets the connection configuration of the BMS server.

Usage

```
set_qbms_config(
  url = "http://localhost/ibpworkbench/controller/auth/login",
  path = NULL,
  page_size = 1000,
  time_out = 120,
  no_auth = FALSE,
  engine = "bms",
  brapi_ver = "v1",
  verbose = TRUE
)
```

Arguments

url	URL of the BMS login page. Default is "http://localhost/ibpworkbench/".
path	API path. Default is NULL.
page_size	Page size. Default is 1000.
time_out	Number of seconds to wait for a response until giving up. Default is 10.
no_auth	TRUE if the server doesn't require authentication/login. Default is FALSE.
engine	Backend database (bms default, breedbase, gigwa, ebs).
brapi_ver	BrAPI version (v1 or v2).
verbose	Logical indicating if progress bar will display on the console when retrieving data from API. TRUE by default.

Value

No return value.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

Examples

```
set_qbms_config("https://bmsdev-brapi.ibp.services/ibpworkbench")
```

`set_qbms_connection` *Set the QBMS Connection*

Description

Sets the QBMS connection object to the current environment.

Usage

```
set_qbms_connection(env)
```

Arguments

`env`	A list containing the connection configuration and status to load.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

`get_qbms_connection`

Examples

```
if(interactive()) {
  # Configure your server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Login using your account (interactive mode)
  # You can pass your username and password as parameters (batch mode)
  login_bms()

  # Select a crop by name
  set_crop("wheat")

  # Select a breeding program by name
```

```
  set_program("Wheat International Nurseries")

  # Get germplasm data
  df1 <- get_germplasm_data("Jabal")

  # Save the current connection (phenotypic server)
  con1 <- get_qbms_connection()

  # Configure QBMS to connect to the genotypic server
  set_qbms_config("https://gigwa.southgreen.fr/gigwa/", engine = "gigwa", no_auth = TRUE)

  # Set the db, project, and run
  gigwa_set_db("DIVRICE_NB")
  gigwa_set_project("refNB")
  gigwa_set_run("03052022")

  # Get associated metadata
  df2 <- gigwa_get_metadata()

  # Save the current connection (before switch)
  con2 <- get_qbms_connection()

  # Load the saved phenotypic server connection
  set_qbms_connection(con1)

  # Continue retrieving germplasm attributes from the phenotypic server
  df3 <- get_germplasm_attributes("Jabal")
}
```

`set_study` *Set the Current Active Study*

Description

Updates the current active study in the internal state object using the studyDbId, which is associated with the given study_name parameter.

Usage

```
set_study(study_name)
```

Arguments

`study_name`	The name of the study.

Value

No return value.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

login_bms, set_crop, set_program, set_trial, list_studies

Examples

```
if(interactive()) {
  # Configure your server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Login using your account (interactive mode)
  # You can pass your username and password as parameters (batch mode)
  login_bms()

  # Select a crop by name
  set_crop("wheat")

  # Select a breeding program by name
  set_program("Wheat International Nurseries")

  # Select a specific study/trial by name
  set_trial("IDYT39")

  # Select a specific environment/location dataset
  set_study("IDYT39 Environment Number 9")
}
```

---

set_token *Set Access Token Response*

Description

If the request for an access token is valid, the authorization server needs to generate an access token and return these to the client, typically along with some additional properties about the authorization.

Usage

```
set_token(token, user = "", expires_in = NULL)
```

Arguments

token	The access token string as issued by the authorization server.
user	The username (optional).
expires_in	The lifetime in seconds of the access token (optional).

Value

No return value.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

`set_trial` *Set the Current Active Trial*

Description

Updates the current active trial in the internal state object using the trialDbId which is associated with the given trial_name parameter.

Usage

```
set_trial(trial_name)
```

Arguments

`trial_name`	The name of the trial.

Value

No return value.

Author(s)

Khaled Al-Shamaa, <k.el-shamaa@cgiar.org>

See Also

`login_bms`, `set_crop`, `set_program`, `list_trials`

Examples

```
if(interactive()) {
  # Configure your server connection
  set_qbms_config("https://bms.icarda.org/ibpworkbench")

  # Login using your account (interactive mode)
  # You can pass your username and password as parameters (batch mode)
  login_bms()

  # Select a crop by name
  set_crop("wheat")

  # Select a breeding program by name
```

```
  set_program("Wheat International Nurseries")

  # Select a specific study/trial by name
  set_trial("IDYT39")
}
```

Index