Package: NSR (via r-universe)

October 17, 2024

Title 'Native Status Resolver'

Version 0.1.0

Description Provides access to the 'Native Status Resolver' (NSR) <https://github.com/ojalaquellueva/nsr> API through R. The user supplies plant taxonomic names and political divisions and the package returns information about their likely native status (e.g., native, non-native,endemic), along with information on how those decisions were made.

Depends R (>= 3.5.0)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports jsonlite, httr

Suggests knitr, rmarkdown, testthat, devtools, BIEN, vcr (>= 0.6.0)

VignetteBuilder knitr

RoxygenNote 7.3.2

NeedsCompilation no

Author Brian Maitner [aut, cre], Brad Boyle [aut]

Maintainer Brian Maitner <`bmaitner@gmail.com`>

Repository CRAN

Date/Publication 2024-10-16 18:00:02 UTC

Contents

`NSR` *Check the native status for plant species in a political region*

Description

NSR returns information on native status for species within a political region.

Usage

```
NSR(occurrence_dataframe, ...)
```

Arguments

`occurrence_dataframe`	A properly formatted dataframe, see https://bien.nceas.ucsb.edu/bien/tools/nsr/batch-mode/
`...`	Additional arguments passed to internal functions.

Value

Dataframe containing NSR results.

Examples

```
## Not run:

results <- NSR(occurrence_dataframe = nsr_testfile)

# Inspect the results
head(results, 10)
# That's a lot of columns. Let's display one row vertically
# to get a better understanding of the output fields
results.t <- t(results[,2:ncol(results)])
results.t[,1,drop =FALSE]
# Summarize the main results
results[ 1:10,
c("species", "country", "state_province", "native_status", "native_status_reason")]

# Compare summary flag isIntroduced to more detailed native_status values
# and inspect souces consulted
results[ 1:10,
c("species", "country", "state_province", "native_status", "isIntroduced", "native_status_sources")]
```

```
## End(Not run)
```

`NSR_citations` *Get citation information*

Description

Returns information needed to cite the NSR

Usage

```
NSR_citations(...)
```

Arguments

`...`	Additional arguments passed to internal functions.

Value

Dataframe containing bibtex-formatted citation information

Note

This function provides citation information in bibtex format that can be used with reference manager software (e.g. Paperpile, Zotero). Please do remember to cite both the sources and the NSR, as the NSR couldn't exist without these sources!

Examples

```
{
citation_info <- NSR_citations()
}
```

`NSR_data_dictionary` *Get NSR data dictionary*

Description

Returns information from the NSR data dictionary

Usage

```
NSR_data_dictionary(native_status = FALSE, ...)
```

Arguments

`native_status`	Logical. If FALSE(Default) returns information on fields. If TRUE, returns information on Native Status categories.
`...`	Additional arguments passed to internal functions.

Value

Data.frame containing requested data dictionary contents.

Examples

```
{
NSR_fields <- NSR_data_dictionary()

status_codes <- NSR_data_dictionary(native_status = TRUE)
}
```

`NSR_metadata` *Get NSR metadata*

Description

Returns metadata on NSR including version and citation information

Usage

```
NSR_metadata(bibtex_file = NULL, ...)
```

Arguments

`bibtex_file`	Optional output file for writing bibtex citations.
`...`	Additional arguments passed to internal functions.

Value

List containing: (1) bibtex-formatted citation information, (2) information about NSR data sources, and (3) NSR version information.

Note

This function provides citation information in bibtex format that can be used with reference manager software (e.g. Paperpile, Zotero). Please remember to cite both the sources and the NSR, as the NSR couldn't exist without these sources!

This function is a wrapper that returns the output of the functions NSR_citations, NSR_sources, and NSR_version.

Examples

```
{
metadata <- NSR_metadata()
}
```

`NSR_political_divisions`

Get information on political divisions with checklists within the NSR

Description

NSR_political_divisions returns information on political divisions with checklist information present in the NSR.

Usage

```
NSR_political_divisions(by_country = TRUE, ...)
```

Arguments

`by_country`	Logical. If TRUE (the default), will return a data.frame of checklists for each country. If FALSE, will return a data.frame of countries for each checklist.
`...`	Additional parameters passed to internal functions.

Value

data.frame containing information on political divisions within the NSR database.

Note

Setting checklist to FALSE returns a list of political divisions that can be used to standardize spellings.

Examples

```
## Not run:

#To get a list of all political divisions with comprehensive checklists:
checklists_per_country <- NSR_political_divisions()

#To get a list of all checklists the associated countries, set "by_country" to FALSE
countries_per_checklist <- NSR_political_divisions(by_country=FALSE)


## End(Not run)
```

`NSR_simple` *Check the native status for plant species in a political region*

Description

NSR_simple returns information on native status for species within a political region.

Usage

```
NSR_simple(
  species = NULL,
  country = NULL,
  state_province = NULL,
  county_parish = NULL,
  ...
)
```

Arguments

`species`	A single species or a vector of species, with genus and specific epithet separated by a space.
`country`	A single country or a vector of countries. If a vector, length must equal length of species vector.
`state_province`	A single state/province or a vector of states. If a vector, length must equal length of species vector.
`county_parish`	A single county/parish or a vector of counties. If a vector, length must equal length of species vector.
`...`	Additional arguments passed to internal functions.

Value

Dataframe containing NSR results.

Examples

```
## Not run:

results <- NSR_simple(species = "Acer rubrum",
           country = "Canada",state_province = "Ontario")

results <- NSR_simple(species = c("Acer rubrum", "Aspen tremuloides") ,
           country = c("Canada","Canada"),state_province = c("Ontario","Ontario"))


## End(Not run)
```

`NSR_sources` *Get information on sources used by the NSR*

Description

Return metadata about the current NSR sources

Usage

```
NSR_sources(...)
```

Arguments

`...`	Additional arguments passed to internal functions.

Value

Dataframe containing information about the sources used in the current NSR version.

Examples

```
{
sources <- NSR_sources()
}
```

`NSR_template` *Make a template for an NSR query*

Description

NSR_template builds a template that can be populated to submit an NSR query.

Usage

```
NSR_template(nrow = 1)
```

Arguments

`nrow`	The number of rows to include in the template

Value

Template data.frame that can be populated and then used in NSR queries.

Examples

```
## Not run:

template<-NSR_template(nrow = 2)
template$genus<-"Acer"
template$species<-c("Acer rubrum", "Acer saccharum")
template$country<-"Canada"
template$user_id<-1:2
results <- NSR(occurrence_dataframe = template)


## End(Not run)
```

`nsr_testfile` *Example NSR data*

Description

A sample dataset showing the proper formatting of NSR inputs.

Usage

```
nsr_testfile
```

Format

A data.frame with 22 observations of 5 variables:

taxon taxon, "Genus species" format, with specific epithet optional

country country that a species occurs within

state_province state/province that a species occurs within

county_parish county/parish that a species occurs within

user_id unique numeric ID that can be used to link to original dataset ...

Source

https://biendata.org

`NSR_version` *Get metadata on current NSR version*

Description

Return metadata about the current NSR version

Usage

```
NSR_version(...)
```

Arguments

`...`	Additional arguments passed to internal functions.

Value

Dataframe containing current NSR version number, build date, and code version.

Examples

```
{
NSR_version_metadata <- NSR_version()
}
```

Index