Package 'wdpar'

May 8, 2026

Type Package

Version 1.3.9

Title Interface to the World Database on Protected Areas

Description Fetch and clean data from the World Database on Protected Areas (WDPA) and the World Database on Other Effective Area-Based Conservation Measures (WDOECM). Data is obtained from Protected Planet <https://www.protectedplanet.net/en>. To augment data cleaning procedures, users can install the 'prepr' R package (available at <https://github.com/prioritizr/prepr>). For more information on this package, see Hanson (2022) <doi:10.21105/joss.04594>.

Imports utils, assertthat (>= 0.2.0), progress (>= 1.2.0), curl (>= 3.2), httr (>= 1.3.1), countrycode (>= 1.1.0), chromote (>= 0.2.0), xml2 (>= 1.2.0), cli (>= 1.0.1), lwgeom (>= 0.2-1), tibble (>= 2.1.3), pingr (>= 1.1.2), rappdirs (>= 0.3.1), withr (>= 3.0.0), archive (>= 1.1.10)

Suggests testthat (>= 2.0.1), knitr (>= 1.2.0), roxygen2 (>= 6.1.1), rmarkdown (>= 1.10), ggmap (>= 4.0.0), ggplot2 (>= 3.1.0), prepr (>= 0.3.0), dplyr (>= 1.0.7), ps (>= 1.5.0)

Depends R (>= 3.5.0), sf (>= 1.0-13)

License GPL-3

Encoding UTF-8

Language en-US

URL https://prioritizr.github.io/wdpar/, https://github.com/prioritizr/wdpar

BugReports https://github.com/prioritizr/wdpar/issues

VignetteBuilder knitr

RoxygenNote 7.3.3

Collate 'internal.R' 'package.R' 'read_sf_n.R' 'st_erase_overlaps.R' 'st_repair_geometry.R' 'wdpa_clean.R' 'wdpa_dissolve.R' 'wdpa_url.R' 'wdpa_latest_version.R' 'wdpa_fetch.R' 'wdpa_read.R' 'zzz.R'

NeedsCompilation no

Author Jeffrey O Hanson [aut, cre]

Maintainer Jeffrey O Hanson <jeffrey.hanson@uqconnect.edu.au>

Repository CRAN

Date/Publication 2026-01-26 09:50:02 UTC

Contents

`st_erase_overlaps` *Erase overlaps*

Description

Erase overlapping geometries in a `sf::sf()` object.

Usage

```
st_erase_overlaps(x, verbose = FALSE)
```

Arguments

`x`	`sf::sf()` object.
`verbose`	`logical` should progress be reported? Defaults to `FALSE`.

Details

This is a more robust – albeit slower – implementation for `sf::st_difference()` when `y` is missing.

Value

A `sf::sf()` object.

See Also

sf::st_difference(), wdpa_dissolve().

Examples

```
# create data
pl1 <- sf::st_polygon(list(matrix(c(0, 0, 2, 0, 1, 1, 0, 0), byrow = TRUE,
                                  ncol = 2))) * 100
pl2 <- sf::st_polygon(list(matrix(c(0, 0.5, 2, 0.5, 1, 1.5, 0, 0.5),
                                  byrow = TRUE, ncol = 2))) * 100
pl3 <- sf::st_polygon(list(matrix(c(0, 1.25, 2, 1.25, 1, 2.5, 0, 1.25),
                                  byrow = TRUE, ncol = 2))) * 100
x <- sf::st_sf(order = c("A", "B", "C"),
               geometry = sf::st_sfc(list(pl1, pl2, pl3), crs = 3395))

# erase overlaps
y <- st_erase_overlaps(x)

# plot data for visual comparison
par(mfrow = c(1, 2))
plot(sf::st_geometry(x), xlim = c(0, 200), ylim = c(0, 250),
     main = "original", col = "transparent")
plot(sf::st_geometry(y), , xlim = c(0, 200), ylim = c(0, 250),
     main = "no overlaps", col = "transparent")
```

---

st_repair_geometry *Repair geometry*

Description

Repair the geometry of a sf::st_sf() object.

Usage

```
st_repair_geometry(x, geometry_precision = 1500)
```

Arguments

x	sf::sf() object.
geometry_precision	numeric level of precision for processing the spatial data (used with sf::st_set_precision()). The default argument is 1500 (higher values indicate higher precision). This level of precision is generally suitable for analyses at the national-scale. For analyses at finer-scale resolutions, please consider using a greater value (e.g. 10000).

Details

This function works by first using the sf::st_make_valid() function to attempt to fix geometry issues. Since the sf::st_make_valid() function sometimes produce incorrect geometries in rare cases (e.g. when fixing invalid geometries that cross the dateline), this function then uses the st_prepair() function from the **prepr** package to fix those geometries instead (see https://github.com/prioritizr/prepr for details).

Value

A sf::sf() object.

Installation

This function uses the **prepr** package to help repair geometries in certain cases. Because the **prepr** package is not available on the Comprehensive R Archive Network (CRAN), it must be installed from its online code repository. To achieve this, please use the following code:

```
if (!require(remotes)) install.packages("remotes")
remotes::install_github("prioritizr/prepr")
```

Note that the **prepr** package has system dependencies that need to be installed before the package itself can be installed (see package README file for platform-specific instructions).

Examples

```
# create sf object
p1 <- st_sf(
  id = 1,
  geometry = st_as_sfc("POLYGON((0 0, 0 10, 10 0, 10 10, 0 0))", crs = 3857)
)

# repair geometry
p2 <- st_repair_geometry(p1)

# print object
print(p2)
```

---

wdpar *wdpar: Interface to the World Database on Protected Areas*

Description

The **wdpar** R package provides an interface to data provided by Protected Planet. Specifically, it can be used to automatically obtain data from the World Database on Protected Areas (WDPA) and the World Database on Other Effective Area-Based Conservation Measures (WDOECM). It also provides methods for cleaning data from these databases following best practices (outlined in Butchart *et al.* 2015; Protected Planet 2021; Runge *et al.* 2015). The main functions are wdpa_fetch() for downloading data and wdpa_clean() for cleaning data. For more information, please see the package vignette. To cite this package, please see citation("wdpar").

Author(s)

Maintainer: Jeffrey O Hanson <jeffrey.hanson@uqconnect.edu.au>

References

Butchart SH, Clarke M, Smith RJ, Sykes RE, Scharlemann JP, Harfoot M, ... & Brooks TM (2015) Shortfalls and solutions for meeting national and global conservation area targets. *Conservation Letters*, **8**: 329–337.

Protected Planet (2021) Calculating protected and OECM area coverage. Available at: https://www.protectedplanet.net/en/resources/calculating-protected-area-coverage.

Runge CA, Watson JEM, Butchart HM, Hanson JO, Possingham HP & Fuller RA (2015) Protected areas and global conservation of migratory birds. *Science*, **350**: 1255–1258.

See Also

Useful links:

- https://prioritizr.github.io/wdpar/
- https://github.com/prioritizr/wdpar
- Report bugs at https://github.com/prioritizr/wdpar/issues

---

wdpa_clean *Clean data*

Description

Clean data obtained from Protected Planet. Specifically, this function is designed to clean data obtained from the World Database on Protected Areas (WDPA) and the World Database on Other Effective Area-Based Conservation Measures (WDOECM). For recommended practices on cleaning large datasets (e.g. datasets that span multiple countries or a large geographic area), please see below.

Usage

```
wdpa_clean(
  x,
  crs = "ESRI:54017",
  exclude_unesco = TRUE,
  retain_status = c("Designated", "Inscribed", "Established"),
  snap_tolerance = 1,
  simplify_tolerance = 0,
  geometry_precision = 1500,
  erase_overlaps = TRUE,
  repair_geometries = TRUE,
  verbose = interactive()
)
```

Arguments

x	sf::sf() object containing protected area data.
crs	character or integer object representing a coordinate reference system. Defaults to World Behrmann ("ESRI:54017").
exclude_unesco	logical should UNESCO Biosphere Reserves be excluded? Defaults to TRUE.
retain_status	character vector containing the statuses for protected areas that should be retained during the cleaning process. Available statuses include: "Proposed", "Inscribed", "Adopted", "Designated", and "Established". Additionally, a NULL argument can be specified to ensure that no protected areas are excluded according to their status. The default argument is a character vector containing "Designated", "Inscribed", and "Established". This default argument ensures that protected areas that are not currently implemented are excluded.
snap_tolerance	numeric tolerance for snapping geometry to a grid for resolving invalid geometries. Defaults to 1 meter.
simplify_tolerance	numeric simplification tolerance. Defaults to 0 meters.
geometry_precision	numeric level of precision for processing the spatial data (used with sf::st_set_precision()). The default argument is 1500 (higher values indicate higher precision). This level of precision is generally suitable for analyses at the national-scale. For analyses at finer-scale resolutions, please consider using a greater value (e.g. 10000).
erase_overlaps	logical should overlapping boundaries be erased? This is useful for making comparisons between individual protected areas and understanding their "effective" geographic coverage. On the other hand, this processing step may not be needed (e.g. if the protected area boundaries are going to be rasterized), and so processing time can be substantially by skipping this step and setting the argument to FALSE. Defaults to TRUE.
repair_geometries	logical indicating if the data cleaning procedure should repair invalid spatial geometries? Although it is often necessary to repair geometries when working with national or multi-national scales, it may introduce, spatial artifacts that cause problems at local scales. Defaults to TRUE. Also, please be aware that setting repair_geometries = FALSE may result in errors during the data cleaning process.
verbose	logical should progress on data cleaning be reported? Defaults to TRUE in an interactive session, otherwise FALSE.

Details

This function cleans data following best practices (Butchart *et al.* 2015; Protected Planet 2021; Runge *et al.* 2015). To obtain accurate protected area coverage statistics for a country, please note that you will need to manually clip the cleaned data to the countries' coastline and its Exclusive Economic Zone (EEZ).

1. Exclude protected areas according to their status (i.e. "STATUS" field). Specifically, protected areas that have a status not specified in the argument to retain_status are excluded. By

default, only protected areas that have a "Designated", "Inscribed", or "Established" status are retained. This means that the default behavior is to exclude protected that are not currently implemented.

2. Exclude United Nations Educational, Scientific and Cultural Organization (UNESCO) Biosphere Reserves (Coetzer *et al.* 2014). This step is only performed if the argument to exclude_unesco is TRUE.
3. Standardize column names to ensure consistency between data stored in geodatabase and shapefile formats. Specifically, if present, the "PARENT_ISO3" field is renamed to "PARENT_ISO" and the "SHAPE" field is renamed to "geometry". Note that this field was deprecated by Protected Planet in November 2025 and so this step is not performed for more recent versions of the database. This information is now provided by the "PRNT_ISO3" field.
4. Create a field ("GEOMETRY_TYPE") indicating if areas are represented as point localities ("POINT") or as polygons ("POLYGON").
5. Exclude areas represented as point localities that do not have a reported spatial extent (i.e. missing data for the field
6. Geometries are wrapped to the dateline (using sf::st_wrap_dateline() with the options "WRAPDATELINE=YES" and "DATELINEOFFSET=180").
7. Reproject data to coordinate system specified in argument to crs (using sf::st_transform()).
8. Repair any invalid geometries that have manifested (using st_repair_geometry()). Note that this step is not performed if repair_geometries = FALSE.
9. Buffer areas represented as point localities to circular areas using their reported spatial extent (using data in the field "REP_AREA" and sf::st_buffer(); see Visconti *et al.* 2013).
10. Snap the geometries to a grid to fix any remaining geometry issues (using argument to snap_tolerance and lwgeom::st_snap_to_grid()).
11. Repair any invalid geometries that have manifested (using st_repair_geometry()). Note that this step is not performed if repair_geometries = FALSE.
12. Simplify the protected area geometries to reduce computational burden (using argument to simplify_tolerance and sf::st_simplify()). Note that this step is not performed if repair_geometries = FALSE.
13. Repair any invalid geometries that have manifested (using st_repair_geometry()).
14. If the "MARINE" field is present, then it will be converted from integer codes to descriptive names (i.e. 0 = "terrestrial", 1 = "partial", 2 = "marine"). Note that this field was deprecated by Protected Planet in November 2025 and so this step is not performed for more recent versions of the database. This information is now provided by the "REALM" field, wherein sites with over 10% of their area on land are assigned a "Terrestrial" value, sites with over 90% of their area in marine areas are assigned a "Marine" value, and remaining sites are assigned a "Coastal"‘ value.
15. If the "PA_DEF" field is present, then it will be converted from integer codes to descriptive names (i.e. 0 = "OECM", and 1 = "PA"). Note that this field was deprecated by Protected Planet in November 2025 and so this step is not performed for more recent versions of the database. This information is now provided by the "SITE_TYPE" field.
16. Zeros in the "STATUS_YR" field are replaced with missing values (i.e. NA_real_ values).
17. Zeros in the "NO_TK_AREA" field are replaced with NA values for areas where such data are not reported or applicable (i.e. areas with the values "Not Applicable" or "Not Reported" in the "NO_TK_AREA" field).

18. Overlapping geometries are erased from the protected area data (discussed in Deguignet *et al.* 2017). Geometries are erased such that areas associated with more effective management categories (`"IUCN_CAT"`) or have historical precedence are retained (using `sf::st_difference()`).
19. Slivers are removed (geometries with areas less than 0.1 square meters).
20. The size of areas are calculated in square kilometers and stored in the field `"AREA_KM2"`.
21. Trimming extra leading or trailing white space characters (i.e., `" "`, `"\n"`, `"\r"`) from fields that contain character values.

Value

A `sf::sf()` object.

Recommended practices for large datasets

This function can be used to clean large datasets assuming that sufficient computational resources and time are available. Indeed, it can clean data spanning large countries, multiple countries, and even the full global dataset. When processing the full global dataset, it is recommended to use a computer system with at least 32 GB RAM available and to allow for at least one full day for the data cleaning procedures to complete. It is also recommended to avoid using the computer system for any other tasks while the data cleaning procedures are being completed, because they are very computationally intensive. Additionally, when processing large datasets – and especially for the global dataset – it is strongly recommended to disable the procedure for erasing overlapping areas. This is because the built-in procedure for erasing overlaps is very time consuming when processing many protected areas, so that information on each protected area can be output (e.g. IUCN category, year established). Instead, when cleaning large datasets, it is recommended to run the data cleaning procedures with the procedure for erasing overlapping areas disabled (i.e. with `erase_overlaps = FALSE`). After the data cleaning procedures have completed, the protected area data can be manually dissolved to remove overlapping areas (e.g. using `wdpa_dissolve()`). For an example of processing a large protected area dataset, please see the vignette.

References

Butchart SH, Clarke M, Smith RJ, Sykes RE, Scharlemann JP, Harfoot M, ... & Brooks TM (2015) Shortfalls and solutions for meeting national and global conservation area targets. *Conservation Letters*, **8**: 329–337.

Coetzer KL, Witkowski ET, & Erasmus BF (2014) Reviewing Biosphere Reserves globally: Effective conservation action or bureaucratic label? *Biological Reviews*, **89**: 82–104.

Deguignet M, Arnell A, Juffe-Bignoli D, Shi Y, Bingham H, MacSharry B & Kingston N (2017) Measuring the extent of overlaps in protected area designations. *PloS One*, **12**: e0188681.

Runge CA, Watson JEM, Butchart HM, Hanson JO, Possingham HP & Fuller RA (2015) Protected areas and global conservation of migratory birds. *Science*, **350**: 1255–1258.

Protected Planet (2021) Calculating protected and OECM area coverage. Available at: https://www.protectedplanet.net/en/resources/calculating-protected-area-coverage.

Visconti P, Di Marco M, Alvarez-Romero JG, Januchowski-Hartley SR, Pressey, RL, Weeks R & Rondinini C (2013) Effects of errors and gaps in spatial data sets on assessment of conservation progress. *Conservation Biology*, **27**: 1000–1010.

See Also

wdpa_fetch(), wdpa_dissolve().

Examples

```
## Not run:
# fetch data for the Liechtenstein
lie_raw_data <- wdpa_fetch("LIE", wait = TRUE)

# clean data
lie_data <- wdpa_clean(lie_raw_data)

# plot cleaned dataset
plot(lie_data)


## End(Not run)
```

---

wdpa_dissolve *Dissolve data*

Description

Create a dataset of spatial boundaries that contains no overlapping geometries.

Usage

```
wdpa_dissolve(x, geometry_precision = 1500)
```

Arguments

x	sf::sf() object.
geometry_precision	numeric level of precision for processing the spatial data (used with sf::st_set_precision()). The default argument is 1500 (higher values indicate higher precision). This level of precision is generally suitable for analyses at the national-scale. For analyses at finer-scale resolutions, please consider using a greater value (e.g. 10000).

Details

This function is basically a wrapper for sf::st_union(). It also contains additional parameters to assist with processing large and complex geometry data.

Value

A sf::sf() object.

See Also

sf::st_union(), st_erase_overlaps().

Examples

```
# create data
pl1 <- sf::st_polygon(list(matrix(c(0, 0, 2, 0, 1, 1, 0, 0), byrow = TRUE,
                                  ncol = 2))) * 100
pl2 <- sf::st_polygon(list(matrix(c(0, 0.5, 2, 0.5, 1, 1.5, 0, 0.5),
                                  byrow = TRUE, ncol = 2))) * 100
pl3 <- sf::st_polygon(list(matrix(c(0, 1.25, 2, 1.25, 1, 2.5, 0, 1.25),
                                  byrow = TRUE, ncol = 2))) * 100
x <- sf::st_sf(order = c("A", "B", "C"),
               geometry = sf::st_sfc(list(pl1, pl2, pl3), crs = 3395))

# dissolve data
y <- wdpa_dissolve(x)

# plot data for visual comparison
par(mfrow = c(1, 2))
plot(sf::st_geometry(x), xlim = c(0, 200), ylim = c(0, 250),
     main = "original", col = "transparent")
plot(sf::st_geometry(y), , xlim = c(0, 200), ylim = c(0, 250),
     main = "dissolved", col = "transparent")
```

---

wdpa_fetch *Fetch data*

Description

Fetch data from Protected Planet. Specifically, data are downloaded from the World Database on Protected Areas (WDPA) and the World Database on Other Effective Area-Based Conservation Measures (WDOECM). **Note that data are downloaded assuming non-commercial use.**

Usage

```
wdpa_fetch(
  x,
  wait = FALSE,
  download_dir = tempdir(),
  force_download = FALSE,
  check_version = TRUE,
  n = NULL,
  page_wait = 2,
  datatype = "gdb",
  verbose = interactive()
)
```

Arguments

x	character country for which to download data. This argument can be the name of the country (e.g. "Liechtenstein") or the ISO-3 code for the country (e.g. "LIE"). This argument can also be set to "global" to download all of the protected areas available in the database (approximately 1.1 GB).
wait	logical if data is not immediately available for download should the session be paused until it is ready for download? If argument to wait is FALSE and the data is not ready then NA will be returned. Defaults to FALSE.
download_dir	character folder path to download the data. Defaults to a temporary directory. To avoid downloading the same dataset multiple times, it is recommended to use a persistent directory (e.g. rappdirs::user_data_dir("wdpar"); see Examples below).
force_download	logical if the data has previously been downloaded and is available at argument to download_dir, should a fresh copy be downloaded? Defaults to FALSE.
check_version	logical if the data are being imported from from the argument to download_dir, should the data be checked to see if the version number matches the latest version available online? Defaults to TRUE.
n	integer number of records to import per data source. Defaults to NULL such that all data are imported.
page_wait	numeric number of seconds to wait for web pages to load when finding the download URL on Protected Planet. Defaults to 2. Since the process of finding a download URL requires navigating through multiple web pages, the default argument means that the function will take at least 8 seconds to complete. Users on slow internet connections may experience issues with the default argument (e.g. resulting in an error containing the message Error: Summary: NoSuchElement). To avoid this, users can try specifying a greater value (e.g. 5 seconds).
datatype	character denoting the file format for which to download protected area data. Available options include: ("shp") shapefile format and ("gdb") file geodatabase format. Defaults to '"gdb". Note that global data are only available in file geodatabase format.
verbose	logical should a progress on downloading data be reported? Defaults to TRUE in an interactive session, otherwise FALSE.

Details

This function obtains and imports data from Protected Planet. By default (per force_download = FALSE), it will check to see if the data have already been downloaded and, if so, simply import the previously downloaded data. It will also check to see if a newer version of the dataset is available on Protected Planet (per check_version = TRUE) and, if so, provide an alert. If the latest version is not required, this alert can be safely ignored. However, if the latest version of the data is required, then using force_download = TRUE will ensure that the latest version is always obtained. After importing the data, it is strongly recommended to clean the data prior to analysis (see wdpa_clean()).

Value

A sf::sf() object.

Data source

The PA_DEF column indicates the data source for individual areas and sites that comprise the imported dataset. Specifically, data obtained through the World Database on Protected Areas (WDPA) are indicated with a value of 1 in the PA_DEF column. Additionally, data obtained through the World Database on Other Effective Area-Based Conservation Measures (WDOECM) are indicated with a value of 0 in the PA_DEF column. For more details on data conventions, please consult the official manual (UNEP-WCMC 2019).

Troubleshooting

The function requires a Chromium-based browser (e.g., Google Chrome, Chromium, or Brave) to be installed. This is because it uses the **chromote** to find the URL for downloading data from Protected Planet. If you don't have one of these browsers installed, then please try installing Google Chrome. If you do have one of these browsers installed and this function throws an error indicating that it can't find the browser, try setting the CHROMOTE_CHROME environment variable to the file path of the executable. For example, you could do this with:

```
Sys.setenv(CHROMOTE_CHROME = "INSERT_FILE_PATH_HERE.exe")
```

Also, the function will sometimes produce messages that complain about handle_read_frame or unpromised promise errors. Please understand that these messages are, in fact, not errors and can be safely ignored (see https://github.com/rstudio/chromote/pull/111). As such, if you see these messages when running the function, you can assume that the function has still worked correctly. For reference, the misleading messages will look something like the following:

```
[error] handle_read_frame error: websocketpp.transport:7 (End of File)
Unhandled promise error: Chromote: timed out waiting for response to command Browser.close
```

For further help with troubleshooting, please refer to the documentation for the **chromote** package (https://rstudio.github.io/chromote/).

References

UNEP-WCMC (2019). User Manual for the World Database on Protected Areas and world database on other effective area-based conservation measures: 1.6. UNEP-WCMC: Cambridge, UK. Available at: https://wcmc.io/WDPA_Manual.

See Also

wdpa_clean(), wdpa_read(), wdpa_url(), countrycode::countrycode().

Examples

```
## Not run:
# fetch data for Liechtenstein
lie_raw_data <- wdpa_fetch("Liechtenstein", wait = TRUE)

# print data
print(lie_raw_data)
```

```
# plot data
plot(lie_raw_data)

# fetch data for Liechtenstein using the ISO3 code
lie_raw_data <- wdpa_fetch("LIE", wait = TRUE)

# since data are saved in a temporary directory by default,
# a persistent directory can be specified to avoid having to download the
# same dataset every time the R session is restarted
lie_raw_data <- wdpa_fetch("LIE", wait = TRUE,
                           download_dir = rappdirs::user_data_dir("wdpar"))

# data for multiple countries can be downloaded separately and combined,
# this is useful to avoid having to download the global dataset
## load packages to easily merge datasets
library(dplyr)
library(tibble)

## define country names to download
country_codes <- c("LIE", "MHL")

## download data for each country
mult_data <- lapply(country_codes, wdpa_fetch, wait = TRUE)

## merge datasets together
mult_dat <- st_as_sf(as_tibble(bind_rows(mult_data)))

## print data
print(mult_dat)

## End(Not run)
```

`wdpa_latest_version` *Query latest version*

Description

Find the latest version of the combined World Database on Protected Areas (WDPA) and World Database on Other Effective Area-Based Conservation Measures (WDOECM) dataset. This is a character identifier representing the month and year (e.g. `Sep2020`) the data were released.

Usage

```
wdpa_latest_version()
```

Details

The version number is determined using a web address where the global dataset is available. For specific details, please refer to the source code for this function.

Value

A character value with the dataset version.

Examples

```
## Not run:
# find the latest version
wdpa_latest_version()

## End(Not run)
```

---

wdpa_read *Read data*

Description

Read data obtained from Protected Planet. Specifically, this function is designed to import data obtained from the World Database on Protected Areas (WDPA) and the World Database on Other Effective Area-Based Conservation Measures (WDOECM).

Usage

```
wdpa_read(x, n = NULL)
```

Arguments

x	character file name for a zip archive file downloaded from https://www.protectedplanet.net/en.
n	integer number of records to import per data source. Defaults to NULL such that all data are imported.

Details

This function assumes that data have previously been downloaded to your computer, and need to import the data. After importing the data, it is strongly recommended to clean the data prior to analysis (see wdpa_clean()).

Value

A sf::sf() object.

Data source

The PA_DEF column indicates the data source for individual areas and sites that comprise the imported dataset. Specifically, data obtained through the World Database on Protected Areas (WDPA) are indicated with a value of 1 in the PA_DEF column. Additionally, data obtained through the World Database on Other Effective Area-Based Conservation Measures (WDOECM) are indicated with a value of 0 in the PA_DEF column. For more details on data conventions, please consult the official manual (UNEP-WCMC 2019).

References

UNEP-WCMC (2019). User Manual for the World Database on Protected Areas and world database on other effective area-based conservation measures: 1.6. UNEP-WCMC: Cambridge, UK. Available at: https://wcmc.io/WDPA_Manual.

See Also

wdpa_fetch(), wdpa_clean().

Examples

```
## Not run:
# find url for Liechtenstein dataset
download_url <- wdpa_url("LIE", wait = TRUE)

# path to save file zipfile with data
path <- tempfile(pattern = "WDPA_", fileext = ".zip")

# download zipfile
result <- httr::GET(download_url, httr::write_disk(path))

# load data
lie_raw_data <- wdpa_read(path)

# plot data
plot(lie_raw_data)

## End(Not run)
```

---

wdpa_url *Download URL*

Description

Obtain a URL to download data from Protected Planet. Specifically, the URL provides access to data available through the World Database on Protected Areas (WDPA) and the World Database on Other Effective Area-Based Conservation Measures (WDOECM). **Note that data are accessed assuming non-commercial use.**

Usage

```
wdpa_url(x, wait = FALSE, page_wait = 2, datatype = "gdb")
```

Arguments

x	character country for desired data. This argument can be the name of the country (e.g. "Liechtenstein") or the ISO-3 code for the country (e.g. "LIE"). This argument can also be set to "global" to obtain the URL for the global dataset.
wait	logical if data is not immediately available for download should the session be paused until it is ready for download? If argument to wait is FALSE and the data is not ready then an error will be thrown. Defaults to FALSE.
page_wait	numeric number of seconds to wait for web pages to load when finding the download URL on Protected Planet. Defaults to 2. Since the process of finding a download URL requires navigating through multiple web pages, the default argument means that the function will take at least 8 seconds to complete. Users on slow internet connections may experience issues with the default argument (e.g. resulting in an error containing the message Error: Summary: NoSuchElement). To avoid this, users can try specifying a greater value (e.g. 5 seconds).
datatype	character denoting the file format for which to download protected area data. Available options include: ("shp") shapefile format and ("gdb") file geodatabase format. Defaults to '"gdb". Note that global data are only available in file geodatabase format.

Value

A character value with the URL to download the data.

See Also

wdpa_fetch(), countrycode::countrycode().

Examples

```
## Not run:
# obtain url for New Zealand data
nzl_url <- wdpa_url("New Zealand", wait = TRUE)
print(nzl_url)

# obtain url for New Zealand data using its ISO3 code
nzl_url <- wdpa_url("NZL", wait = TRUE)
print(nzl_url)

# obtain url for global data
global_url <- wdpa_url("global")
print(global_url)

## End(Not run)
```

Index