Package: rsat (via r-universe)

August 29, 2024

Type Package

Title Dealing with Multiplatform Satellite Images

Version 0.1.21

Maintainer Unai Pérez - Goya <unai.perez@unavarra.es>

Description Downloading, customizing, and processing time series of satellite images for a region of interest. 'rsat' functions allow a unified access to multispectral images from Landsat, MODIS and Sentinel repositories. 'rsat' also offers capabilities for customizing satellite images, such as tile mosaicking, image cropping and new variables computation. Finally, 'rsat' covers the processing, including cloud masking, compositing and gap-filling/smoothing time series of images (Militino et al., 2018 <doi:10.3390/rs10030398> and Militino et al., 2019 <doi:10.1109/TGRS.2019.2904193>).

Depends R (>= 4.3.0)

Imports XML, curl, httr, leafem, leaflet, rjson, rvest, tmap, xml2, zip, methods, Rdpack, fields, calendR, sf, stars, terra, sp, raster

URL https://docs.ropensci.org/rsat/, https://github.com/ropensci/rsat

BugReports https://github.com/ropensci/rsat/issues

RdMacros Rdpack

Suggests knitr, rmarkdown, covr, testthat

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Collate 'add2rtoi.R' 'api_dataspace.R' 'api_lpdaac.R' 'api_usgs.R' 'connections.R' 'data.R' 'extent_crs.R' 'mosaic_fun_SYN.R' 'mosaic_fun_ls.R' 'mosaic_fun_mod.R' 'mosaic_fun_sen2.R' 'mosaic_generic.R' 'package_tools.R' 'records.R' 'rtoi.R' 'plot.R' 'rsat.R' 'rsat_cloud_mask.R' 'rsat_derive.R'

'rsat_download.R' 'rsat_list_data.R' 'rsat_mosaic.R'
'rsat_preview.R' 'rsat_search.R' 'rsat_smoothing_images.R'
'testing_function.R' 'variables.R'

VignetteBuilder knitr

Repository https://ropensci.r-universe.dev

RemoteUrl https://github.com/ropensci/rsat

RemoteRef master

RemoteSha 4e9e6caa103a77c91dfaba2e309f6bf351b4f967

Contents

`as.data.frame,extent_crs-method`
Coerce to a Data Frame

Description

Functions to check if an object is a data frame, or coerce it if possible.

Usage

```
## S4 method for signature 'extent_crs'
as.data.frame(x)

## S4 method for signature 'records'
as.data.frame(x)
```

Arguments

x	Any R object.

Value

returns a data frame, normally with all row names

Examples

```
## Not run:
# load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

# get the records
rcds <- records(navarre)
# coerce the records to rtoi
df <- as.data.frame(rcds)
# print the dataframe
print(df)

## End(Not run)
```

`as.records` *Create records object from data frame*

Description

Create records object from data frame

Usage

```
as.records(x)

## S4 method for signature 'data.frame'
as.records(x)
```

Arguments

x	a `data.frame` with columns representing the slots of records.

Value

returns a records objects with the columns values in `x`

Examples

```
## Not run:
# load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

navarre <- read_rtoi(file.path(tempdir(),"Navarre"))
```

```
# get the records
rcds <- records(navarre)
# coerce the records to dataframr
df <- as.data.frame(rcds)
# print the dataframe
print(df)

# coerce the dataframe to records
rcds2 <- as.records(df)
# check the conversion
identical(rcds,rcds2)

## End(Not run)
```

`c,extent_crs-method` *Combine values into a vector or a list*

Description

This is a generic function which combines its arguments.

Usage

```
## S4 method for signature 'extent_crs'
c(x, ...)

## S4 method for signature 'records'
c(x, ...)
```

Arguments

`x`	a `records` object.
`...`	additional arguments.

Details

The default method combines its arguments to form a vector. All arguments are coerced to a common type which is the type of the returned value. All attributes except names are removed.

Value

a combination of 'x' class elements

dates *Get/set the dates from a* `records` *or an* `rtoi`

Description

Get/set the dates from a `records` or an `rtoi`

Usage

```
dates(x)

## S4 method for signature 'records'
dates(x)

dates(x) <- value

## S4 replacement method for signature 'records'
dates(x) <- value

## S4 method for signature 'rtoi'
dates(x)
```

Arguments

x	a `records` or an `rtoi` object.
value	the new date to asign

Value

returns a vector of `Date` class

Examples

```
## Not run:
# load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

# get a vector of dates includes in rtoi
dates(navarre)

# get the records
rcds <- records(navarre)

# coerce the records to dataframr
```

```
dates(rcds)

## End(Not run)
```

`ex.dem.navarre` *A Digital Elevation Model (DEM) of the region of Navarre (Spain)*

Description

Geographically projected `RasterStack` with the digital elevation model (DEM) of the region of Navarre (Spain). The DEM was obtained from the National Center for Geographic Information of Spain. The DEM is used as a covariate in the Image Mean Anomaly (IMA) algorithm (`rsat_smoothing_images`).

Format

The `RasterStack` contains 6 layers with the same DEM, one for every image in `ex.ndvi.navarre`.

The `RasterStack` coordinates are in the Sinusoidal projection.

name layer names contain the capturing date of the corresponding image in the format "`YYYYJJJ`".

size 113 rows by 105 columns and 6 layers.

`ex.madrid` *A polygon with the border of Madrid (Spain)*

Description

Spatial feature (`sf`) representing the border of Madrid with coordinates in the longitude/latitude format.

`ex.manhattan` *A polygon with the border of Manhattan (USA)*

Description

Spatial feature (`sf`) representing the border of Manhattan with coordinates in the NAD83 format.

`ex.navarre` *A polygon with the border of Navarre (Spain)*

Description

Spatial feature (`sf`) representing the border of Navarre with coordinates in the longitude/latitude format.

ex.ndvi.navarre *A time series of NDVI in Navarre (Spain)*

Description

Geographically projected `RasterBrick` object of the normalized difference vegetation index (NDVI) in Navarre.

Format

The `RasterBrick` contains 6 images, from the 2nd to the 4th of August in 2017 and 2018. The `RasterBrick` coordinates are in the Sinusoidal projection:

name layer names contain the date of the image in the format `"YYYYJJJ"`.

size each layer contains 113 rows and 105 columns.

get_api_name *Get the API name of a* `records`

Description

A function to get or set the api names of an object.

Usage

```
get_api_name(x)

## S4 method for signature 'records'
get_api_name(x)
```

Arguments

`x`	a `records` object.

Value

a character vector containing the API names of the elements in `x`.

Examples

```
## Not run:
# load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

# get the records
rcds <- records(navarre)

# get a vector with the api name of each records
get_api_name(rcds)

## End(Not run)
```

`get_database` *Extracts or assign the path of the database*

Description

Extracts the path to the database from an rtoi/package environment. If both, environment and rtoi database are defined the rtoi database is used.

Usage

```
get_database(x)

## S4 method for signature 'rtoi'
get_database(x)

## S4 method for signature 'missing'
get_database()

set_database(x, ...)

## S4 method for signature 'rtoi'
set_database(x, value)

## S4 method for signature 'character'
set_database(x)
```

Arguments

`x`	an rtoi object.
`...`	additional arguments.
`value`	character argument. The value for change the database directory of `x`.

Value

the database path of an rtoi

Examples

```
## Not run:
# load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

# get the databse used by navarre
get_database(navarre)

# set the a new database path
set_database(navarre,"new_path")

# get the database used by rsat by default
get_database()

# set the a new database path for the entire environment
set_database("new_path")

## End(Not run)
```

---

get_dir — *Get the file path of a* `records` *or an* `rtoi`

Description

Get the file path of a `records` or an `rtoi`

Usage

```
get_dir(x)

## S4 method for signature 'records'
get_dir(x)

## S4 method for signature 'records'
get_order(x)

## S4 method for signature 'rtoi'
get_dir(x)
```

Arguments

x	.

Value

the file path in the records

Examples

```
## Not run:
# load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

# get the path of the
get_dir(navarre)

# get the records
rcds <- records(navarre)

# gets the relative path to store records data
get_dir(rcds)

## End(Not run)
```

---

get_download *Extract the url to download a data record*

Description

It returns a character with the url to download the image.

Usage

```
get_download(x)
```

Arguments

x	a `records` object.

Value

download url of a records

Examples

```
## Not run:
# load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

# get the records
rcds <- records(navarre)
# coerce the records to rtoi
get_download(rcds)

## End(Not run)
```

`get_order` *Get the slot called order from a* `records` *or an* `rtoi`

Description

Get the slot called order from a `records` or an `rtoi`

Usage

```
get_order(x)

get_order(x) <- value

## S4 replacement method for signature 'records'
get_order(x) <- value
```

Arguments

`x`	a `records` or an `rtoi` object.
`value`	logical argument. The new value for `x`.

Value

the value of called order

Examples

```
## Not run:
# load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)
```

```
navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

# get the records
rcds <- records(navarre)

# gets a boolean
get_order(rcds)

## End(Not run)
```

`get_preview` *Extract the url of the preview*

Description

It returns a character vector of urls to preview the data records.

Usage

```
get_preview(x)

## S4 method for signature 'records'
get_preview(x)

## S4 method for signature 'records'
get_download(x)
```

Arguments

x	a `records` object.

Value

preview url of a records

Examples

```
## Not run:
# load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

# get the records
rcds <- records(navarre)
```

```
# get a vector with the preview url of each record
get_api_name(rcds)

## End(Not run)
```

`length,extent_crs-method`

Length of an object

Description

Get or set the length of vectors (including lists) and factors, and of any other R object for which a method has been defined.

Usage

```
## S4 method for signature 'extent_crs'
length(x)

## S4 method for signature 'records'
length(x)
```

Arguments

x	a `records` object to compute its length.

Value

Length currently returns a non-negative integer of length 1

Examples

```
## Not run:
# load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

# get the records
rcds <- records(navarre)

length(rcds)

## End(Not run)
```

names,records-method *Get the name of the object*

Description

A function to get or set the names of an object.

Usage

```
## S4 method for signature 'records'
names(x)

## S4 method for signature 'rtoi'
names(x)

## S4 replacement method for signature 'rtoi,character'
names(x) <- value
```

Arguments

x	a records or an rtoi object.
value	character argument. The new value for x.

Value

a character vector containing the name of all the names in x.

Examples

```
## Not run:
# load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

names(navarre)
names(navarre) <- "New name"
names(navarre)

rcrds <- records(navarre)

names(rcrds)

## End(Not run)
```

new_record *Create a new* records *object*

Description

Create a new records object from scratch

Usage

```
new_record(
  sat,
  name,
  date,
  product,
  download,
  file_path,
  path,
  row,
  tileid,
  preview,
  api_name,
  order,
  extent_crs
)

## S4 method for signature
## 'character,
##   character,
##   Date,
##   character,
##   character,
##   character,
##   numeric,
##   numeric,
##   character,
##   character,
##   character,
##   logical,
##   extent_crs'
new_record(
  sat,
  name,
  date,
  product,
  download,
  file_path,
  path,
```

```
  row,
  tileid,
  preview,
  api_name,
  order,
  extent_crs
)

## S4 method for signature
## 'character,
##   character,
##   Date,
##   character,
##   character,
##   character,
##   numeric,
##   numeric,
##   character,
##   character,
##   character,
##   logical,
##   missing'
new_record(
  sat,
  name,
  date,
  product,
  download,
  file_path,
  path,
  row,
  tileid,
  preview,
  api_name,
  order
)
```

Arguments

sat	the name of the satellite to which the record belongs.
name	the name of the record.
date	the date of the record.
product	the product.
download	the url to download the satellite record.
file_path	the saving directory for the satellite record.
path	the path of the tiling system.
row	the row of the tiling system.

tileid	the tile id.
preview	the url of the preview of the satellite record.
api_name	the api name.
order	boolean, defines if the image must be requested or not.
extent_crs	extent (used to project the preview).

Value

records object

Examples

```
## Not run:
# create a new record from scrach
rcds <- new_record(
  sat = "modis",
  name = "mod09a",
  date = as.Date("2011087", "%Y%j"),
  product = "product",
  download = "url/aaa/download",
  file_path = "file_path",
  path = 1,
  row = 1,
  tileid = "exampleid",
  preview = "url",
  api_name = "nasa_inventory",
  order = FALSE
)
rcds

## End(Not run)
```

---

new_rtoi *Creates a new* rtoi *object*

Description

Creates a new rtoi object

Usage

```
new_rtoi(name, region, rtoi_path, db_path, records, size)

## S4 method for signature 'character,sf,character,missing,missing,missing'
new_rtoi(name, region, rtoi_path)

## S4 method for signature 'character,sf,character,character,missing,missing'
new_rtoi(name, region, rtoi_path, db_path)
```

```
## S4 method for signature 'character,sf,character,character,records,missing'
new_rtoi(name, region, rtoi_path, db_path, records)

## S4 method for signature 'character,sf,character,character,records,numeric'
new_rtoi(name, region, rtoi_path, db_path, records, size)
```

Arguments

name	the name of the region of interest.
region	an sf object.
rtoi_path	the path to the rtoi folder.
db_path	the path to the database.
records	a records object.
size	the size of rtoi folder. By default, the size is computed from rtoi_path.

Value

the reference of the rtoi object

---

plot,rtoi,Date-method *Plot an* rtoi *object*

Description

Plot (a map of) the values of an rtoi or records object.

Usage

```
## S4 method for signature 'rtoi,Date'
plot(
  x,
  y,
  ...,
  variable = "rgb",
  band_name = c("red", "green", "blue"),
  verbose = FALSE,
  xsize = 250,
  ysize = 250
)

## S4 method for signature 'rtoi,character'
plot(
  x,
  y,
  ...,
```

```
  variable = "rgb",
  product = "ALL",
  band_name = c("red", "green", "blue"),
  dates = NULL,
  verbose = FALSE,
  xsize = 250,
  ysize = 250
)

## S4 method for signature 'records,ANY'
plot(x, y, verbose = FALSE, ...)

## S4 method for signature 'rtoi,missing'
plot(x, y, verbose = FALSE, ...)
```

Arguments

x	an rtoi or records.
y	character argument. The valid values are "dates", "preview", or "view".
...	additional arguments.
variable	character argument. The variable to be plotted. By default, a color (RGB) variable is selected .
band_name	character vector argument. Enables false color plots. By default, usual bands are selected c("red","green","blue").
verbose	logical argument. If TRUE, the function prints the running steps and warnings.
xsize	the number of samples on the horizontal axis.
ysize	the number of samples on the vertical axis.
product	character argument. The product name to be plotted.
dates	date vector argument. The dates to be plotted.

Value

tmap plot.

Examples

```
library(rsat)
 ## Not run:

# load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

print(navarre)
```

```
# plot the calendar
plot(navarre, "dates")



# replace with your own "username" and "password"
set_credentials("username", "password")

# plot the quicklook images before the download
# needs credentials to download preview images
plot(navarre, y = "preview")

# select partially cloud free
rcds <- records(navarre)
rcds <- rcds[dates(rcds) %in% as.Date(c("20210310", "20210313"), "%Y%m%d")]
records(navarre) <- rcds

plot(navarre, "preview")

file.copy(from=system.file("ex/Pamplona",package="rsat"),
         to=tempdir(),
         recursive = TRUE)
# plot already mosaicked rtoi ("view" mode)
pamplona <- read_rtoi(file.path(tempdir(),"Pamplona"))

rsat_list_data(pamplona)

# plot can compute the rgb image on the fly from mosaicek bands
plot(pamplona, "view", product="mod09ga")

# plot on the fly with false color
plot(pamplona, "view",
     product = "mod09ga",
     band_name = c("nir", "red", "green"))

file.copy(from=system.file("ex/PamplonaDerived",package="rsat"),
         to=tempdir(),
         recursive = TRUE)
# plot already mosaicked rtoi ("view" mode)
pamplona.derived <- read_rtoi(file.path(tempdir(),"PamplonaDerived"))

rsat_list_data(pamplona.derived)

# plot derived variables
plot(pamplona.derived, "view",
     product = "mod09ga",
     variable = "NDVI")

# Set the max and min value in plot
plot(pamplona.derived,"view",
     variable="NDVI",
     product="mod09ga",
```

```
     zlim=c(0,1))

## End(Not run)
```

`print,api_dataspace-method`
Prints the values

Description

prints an object and returns it invisibly (via invisible(x)).

Usage

```
## S4 method for signature 'api_dataspace'
print(x)

## S4 method for signature 'api_lpdaac'
print(x)

## S4 method for signature 'api_usgs'
print(x)

## S4 method for signature 'extent_crs'
print(x)

## S4 method for signature 'records'
print(x)

## S4 method for signature 'rtoi'
print(x)

## S4 method for signature 'variables'
print(x, ...)
```

Arguments

x	an object to be printed..
...	additional arguments.

Value

prints rtoi metadata

Examples

```
## Not run:
library(rsat)

# load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

print(navarre)

# get records
rcrds <- records(navarre)

print(rcrds)

## End(Not run)
```

`print_credentials` *Prints the credentials for the web services*

Description

Prints the credentials for the web services

Usage

```
print_credentials(...)

## S4 method for signature 'ANY'
print_credentials()
```

Arguments

`...`	additional arguments.

Value

print the credentials asigned in the package environment variable

Examples

```
print_credentials()
set_credentials("example", "example", "earthdata")
print_credentials()
```

product — *Get the name of the product from a* `records` *or an* `rtoi`

Description

Get the name of the product from a `records` or an `rtoi`

Usage

```
product(x)

## S4 method for signature 'records'
product(x)

## S4 method for signature 'rtoi'
product(x)
```

Arguments

`x`	a `records` or an `rtoi` object.

Value

the name of the product in the records

read_rtoi — *Reads an rtoi from the hard drive*

Description

Reads an rtoi from the hard drive

Usage

```
read_rtoi(path, ...)

## S4 method for signature 'character'
read_rtoi(path, ...)
```

Arguments

`path`	an rtoi object.
`...`	additional arguments.

Value

rtoi object readed from disk.

Examples

```
## Not run:
library(rsat)

# load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

navarre <- read_rtoi(file.path(tempdir(),"Navarre"))
print(navarre)

## End(Not run)
```

---

records — *Extracts the satellite records*

Description

returns the object records from an rtoi.

Usage

```
records(x)

## S4 method for signature 'rtoi'
records(x)

records(x) <- value

## S4 replacement method for signature 'rtoi,records'
records(x) <- value
```

Arguments

x	an rtoi object
value	a records object to be set to x.

Value

a set of records in x rtoi

Examples

```
## Not run:
#' library(rsat)
# create a copy of navarre
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)
# load example rtoi
navarre <- read_rtoi(file.path(tempdir(),"Navarre"))
print(navarre)

rcrds <- records(navarre)

records(navarre)<-rcrds[1]
print(navarre)

records(navarre) <- rcrds
print(navarre)
unlink(file.path(tempdir(),"Navarre"),recursive=TRUE)

## End(Not run)
```

`records-class` *A class object for satellite image metadata*

Description

This class object organizes the attributes of satellite images' metadata from several missions/programs uniformly. Structuring the information facilitates managing, previewing, and downloading data records.

Details

`records` works as vector. It accepts usual R methods such as `c`, `[]`, `length()`, `subset()` or `unique()`. Each record (vector element) contains several parameters or slots.

The object can be coerced into a `data.frame` by using the function `as.data.frame()`. The `data.frame` can be transformed back into a `records` with the function `as.records()`.

Slots

`sat` the name of the satellite.

`name` the name of the file.

`date` capturing date of the image.

`product` name of the data product.

`path` the path of the tiling system.

`row` the row of the tiling system.

`tileid` the tile identification number.

`download` the download url.

`file_path` the saving directory for the satellite record.

`preview` the preview url.

`api_name` the name of the API.

`order` boolean, whether the image needs to be ordered.

`extent_crs` coordinate reference system of the preview.

Examples

```
## Not run:
library(rsat)
# create a copy of navarre
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

# load example rtoi
navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

rcrds <- records(navarre)

modis.rcrds <- rcrds[sat_name(rcrds)%in%"Modis"]
ls8.rcrds <- rcrds[sat_name(rcrds)%in%"Landsat-8"]

class(modis.rcrds)
dates(ls8.rcrds)
modis.ls8.records <- c(ls8.rcrds, modis.rcrds)

print(modis.ls8.records)

## End(Not run)
```

`region` *Extracts region from an rtoi*

Description

gets the sf that specifies the region of an rtoi.

Usage

```
region(x)

## S4 method for signature 'rtoi'
region(x)
```

```
region(x) <- value

## S4 replacement method for signature 'rtoi,sf'
region(x) <- value

## S4 replacement method for signature 'rtoi,`NULL`'
region(x) <- value
```

Arguments

x	an rtoi object.
value	an sf object to define the region in x.

Value

the sf class with the region of an rtoi

Examples

```
## Not run:
library(rsat)
# create a copy of navarre
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

# load example rtoi
navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

# get the region from rtoi
sf.obj <-  region(navarre)
plot(sf.obj)

# asign new region value
region(navarre)<-NULL

region(navarre)<-sf.obj

## End(Not run)
```

---

rename *Renames an* rtoi

Description

Renames all parameters and folder name of an rtoi.

Usage

```
rename(x, newname)

## S4 method for signature 'rtoi,character'
rename(x, newname)
```

Arguments

x	an rtoi object
newname	a character class to rename the rtoi.

Value

nothing. the changes the internal name of the rtoi

Examples

```
## Not run:
myrtoi <- read_rtoi("file_path/rtoir_name")
rename(myrtoi, "Navarre_BACK")

## End(Not run)
```

---

rsat *'rsat'*

Description

The goal of 'rsat' is to help you handling time-series of satellite images from multiple platforms in a local, efficient, and standardized way. The package provides tools to;

1. Search (run `vignette("rsat1_search", package = "rsat")`)
2. Download (run `vignette("rsat2_download", package = "rsat")`)
3. Customize, and (run `vignette("rsat3_customize", package = "rsat")`)
4. Process (run `vignette("rsat4_process", package = "rsat")`)

satellite images from Landsat, MODIS, and Sentinel for a region and time of interest.

Registration

The registration in the following online portals is required to get a full access to satellite images with 'rsat';

- USGS USGS is the sole science agency for the Department of the Interior of United States. Provide access to Modis Images. More information about EarthData can be found Here.
- EarthData: A repository of NASA's earth observation data-sets. More information about EarthData can be found here.

- dataspace, a web service giving access to Copernicus' scientific data hub. Please go here to find more details about the data hub.

For convenience, try to use the same username and password for all of them. To satisfy the criteria of all web services make sure that the username is 4 characters long and includes a period, number or underscore. The password must be 12 character long and should include characters with at least one capital letter, and numbers.

Contributing

If you want to contribute by adding new features or fixing bugs in the package you can do it from our github address: https://github.com/spatialstatisticsupna/rsat Bug report: https://github.com/spatialstatisticsupna/rsat/issues

---

rsat_cloudMask *Create cloud mask from an rtoi*

Description

Create cloud mask from an rtoi

Usage

```
rsat_cloudMask(x, ...)

## S4 method for signature 'rtoi'
rsat_cloudMask(x, products = "ALL", verbose = FALSE, overwrite = FALSE, ...)
```

Arguments

x	rtoi object from which cloud masks are computed.
...	additional arguments
products	the name of the dataset from which cloud masks are computed.
verbose	logical argument. If TRUE, the function prints the running steps and warnings.
overwrite	logical argument. If TRUE, overwrites the existing images with the same name.

Value

nothing. The cloud masks will be save in the hard drive. Use get_stars to get the variables.

Examples

```
## Not run:
## Smooth data in rtoi
library(rsat)

# create a copy of pamplona in temp file
file.copy(from=system.file("ex/Pamplona",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

# load example rtoi
pamplona <- read_rtoi(file.path(tempdir(),"Pamplona"))

rsat_cloudMask(pamplona)

rsat_list_data(pamplona)

## End(Not run)
```

`rsat_derive` *Computes a remote sensing index from an* `rtoi`

Description

Combines the bands from multispectral satellite products through simple math to highlight a process or material in the image.

Usage

```
rsat_derive(x, variable, ...)

## S4 method for signature 'rtoi,character'
rsat_derive(
  x,
  variable,
  product,
  dates,
  fun,
  overwrite = FALSE,
  verbose = FALSE,
  suppressWarnings = TRUE,
  ...
)
```

Arguments

`x`	an `rtoi` as the source of images.
`variable`	the name of the variable.

...	additional argument for variable deriving
product	the name of the product from which the index is computed.
dates	a vector of dates being considered (optional).
fun	a function that computes the remote sensing index.
overwrite	logical argument. If TRUE, overwrites the existing images with the same name.
verbose	logical argument. If TRUE, the function prints the running steps and warnings.
suppressWarnings	evaluates its expression in a context that ignores all warnings.

Details

The package contemplates some pre-defined indexes, which can be displayed using the show_variables() function. To compute one of those, write its name in the variable argument. Custom indexes can be supplied through the fun argument. The function should use the name of the bands as inputs (red, green, blue, nir, swir1, or swir2) and return a single element. For instance, the Normalized Difference Snow Index would be;

NDSI = function(green, swir1){ ndsi <- (green - swir1)/(green + swir1) return(ndsi) }

Value

nothing. The derived variables will be save in the hard drive. Use get_stars to get the variables.

Examples

```
## Not run:
library(rsat)

# create a copy of pamplona in temp file
file.copy(from=system.file("ex/Pamplona",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

# load example rtoi
pamplona <- read_rtoi(file.path(tempdir(),"Pamplona"))

rsat_list_data(pamplona)
# show prefedined varibles
show_variables()
rsat_derive(pamplona, "NDVI", product = "mod09ga")
# now NDVI is processed
rsat_list_data(pamplona)

# ad-hoc variable
NDSI = function(green, swir1){
  ndsi <- (green - swir1)/(green + swir1)
  return(ndsi)
}
rsat_derive(pamplona, "NDSI", product = "mod09ga",fun=NDSI)
# now NDVI is processed
```

```
rsat_list_data(pamplona)
plot(pamplona, product="mod09ga",variable="NDSI")

## End(Not run)
```

`rsat_download` *Download the images from a* `records` *or an* `rtoi` *object*

Description

The function saves the raw images in the database or the specified directory. It skips the images that already exist in the database or directory.

Usage

```
rsat_download(x, ...)

## S4 method for signature 'rtoi'
rsat_download(x, db_path, verbose = FALSE, parallel = FALSE, ...)

## S4 method for signature 'records'
rsat_download(x, db_path, verbose = FALSE, parallel = FALSE, ...)
```

Arguments

`x`	a `records` or an `rtoi` object.
`...`	additional arguments.
`db_path`	path to the database. By default, the path is defined by the `rtoi`.
`verbose`	logical argument. If `TRUE`, the function prints the running steps and warnings.
`parallel`	logical argument. If `TRUE`, the function downloads from multiples APIs in parallel.

Value

nothing. Downloads the images into your database

Examples

```
## Not run:
library(rsat)

# create a copy of navarre in temp file
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

# load example rtoi
```

```
navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

# assign the path of the database
set_database(file.path(tempdir(),"DATABASE"))
rsat_download(navarre)

rcrds <-  records(navarre)

rsat_download(rcrds)

## End(Not run)
```

`rsat_get_raster` *Loads into R a time series of images regarding an rtoi, satellite product, and remote sensing index.*

Description

Loads into R a time series of images regarding an rtoi, satellite product, and remote sensing index.

Usage

```
rsat_get_raster(x, p, v, s, ...)

## S4 method for signature 'rtoi'
rsat_get_raster(x, p, v, s, ...)

rsat_get_SpatRaster(x, p, v, s, ...)

## S4 method for signature 'rtoi'
rsat_get_SpatRaster(x, p, v, s, ...)

rsat_get_stars(x, p, v, s, ...)

## S4 method for signature 'rtoi'
rsat_get_stars(x, p, v, s, ...)
```

Arguments

x	an rtoi.
p	a character with the name of the satellite data product.
v	a character with the name of the index.
s	a character with the name of the stage wanted.
...	additional arguments.

Value

a raster stack.

Examples

```
## Not run:
library(rsat)
# load example rtoi
file.copy(from=system.file("ex/PamplonaDerived",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

# load example rtoi
pamplona.derived <- read_rtoi(file.path(tempdir(),"PamplonaDerived"))

# print available variables
rsat_list_data(pamplona.derived)

# get RasterStack from raster package
suppressWarnings(mod.ndvi.raster <-
           rsat_get_raster(pamplona.derived, "mod09ga", "NDVI"))
plot(mod.ndvi.raster)

# get spatraster from terra package
mod.ndvi.rast <- rsat_get_SpatRaster(pamplona.derived, "mod09ga", "NDVI")
plot(mod.ndvi.rast)

# get stars from stars package
suppressWarnings(mod.ndvi.stars <-
rsat_get_stars(pamplona.derived, "mod09ga", "NDVI"))
plot(mod.ndvi.stars)


## get any band in rtoi
# list available data
rsat_list_data(pamplona.derived)
# select band 1: MODIS_Grid_500m_2D_sur_refl_b01_1
mod.ndvi.rast <- rsat_get_SpatRaster(pamplona.derived,
                                     "mod09ga",
                                     "MODIS_Grid_500m_2D_sur_refl_b01_1")
plot(mod.ndvi.rast)

## End(Not run)
```

`rsat_list_data` *List the information available for an* `rtoi`

Description

Displays the existing products, bands, and processing levels for a given `rtoi`

Usage

```
rsat_list_data(x, ...)

## S4 method for signature 'rtoi'
rsat_list_data(x, ...)
```

Arguments

x	an rtoi object.
...	additional arguments.

Value

a data.frame of the available information.

Examples

```
## Not run:
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

# load example rtoi
navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

print(navarre)

# print empty rtoi
rsat_list_data(navarre)

file.copy(from=system.file("ex/Pamplona",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

# load example rtoi
pamplona <- read_rtoi(file.path(tempdir(),"Pamplona"))

print(pamplona)

rtoi.data <- rsat_list_data(pamplona)
# print mosaicked bands
print(rtoi.data)

# print mosaicked bands + derived NDVI
file.copy(from=system.file("ex/PamplonaDerived",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

# load example rtoi
pamplona.derived <- read_rtoi(file.path(tempdir(),"PamplonaDerived"))
rsat_list_data(pamplona.derived)
```

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

`rsat_mosaic` *Mosaic the tiles intersecting the region of interest*

Description

Satellite measurements are divided into indivisible units called tiles. The mosaic function binds and crops the tiles to generate a single image of the region of interest for each date.

Usage

```
rsat_mosaic(x, ...)

## S4 method for signature 'rtoi'
rsat_mosaic(x, ...)

## S4 method for signature 'records'
rsat_mosaic(
  x,
  out_path,
  db_path,
  bfilter = c(),
  warp = "extent",
  region,
  overwrite = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

`x`	an `rtoi` object.
`...`	additional arguments.
`out_path`	path to save the mosaicked images. By default, the path is defined by `x`.
`db_path`	path to the database. By default, the path is defined by `x`.
`bfilter`	a vector of bands to. If not supplied, all are used.
`warp`	character. If equal to "extent", it also crops the images around the `rtoi`. Use "" otherwise.
`region`	an sf object. Region for cropping the images around. By default, the path is defined by `x`.
`overwrite`	logical argument. If `TRUE`, overwrites the existing images with the same name.
`verbose`	boolean. If TRUE show verbose output. Default FALSE

Value

nothing. Mosaics the downloaded images and stored them on the hard disk

Examples

```
## Not run:
library(rsat)

# load navarre sf from the package
data(ex.navarre)

# set the credentials
set_credentials("username", "password")

# path where the region is stored
rtoi.path <- tempdir()
# path where downloads are stored
db.path <- file.path(tempdir(), "DATABASE")
navarre <- new_rtoi(
  "Navarre",
  ex.navarre,
  rtoi.path,
  db.path
) #'
# Landsat-5
rsat_search(
  region = navarre,
  product = "LANDSAT_TM_C1",
  dates = as.Date("1988-08-01") + seq(1, 35)
)
rsat_download(navarre)

rsat_mosaic(navarre, overwrite = T)

rsat_list_data(navarre)

## End(Not run)
```

`rsat_preview` *Preview a* `records` *or an* `rtoi` *object*

Description

Preview a `records` or an `rtoi` object

Usage

```
rsat_preview(x, n, ...)

## S4 method for signature 'rtoi,Date'
rsat_preview(x, n, lpos = c(3, 2, 1), add.layer = FALSE, verbose = FALSE, ...)

## S4 method for signature 'rtoi,missing'
rsat_preview(x, n, lpos = c(3, 2, 1), add.layer = FALSE, verbose = FALSE, ...)

## S4 method for signature 'records,Date'
rsat_preview(
  x,
  n,
  lpos = c(3, 2, 1),
  tmp_dir = file.path(tempdir()),
  add.layer = FALSE,
  verbose = FALSE,
  get.map = TRUE,
  ...
)

## S4 method for signature 'records,numeric'
rsat_preview(
  x,
  n,
  lpos = c(3, 2, 1),
  tmp_dir = file.path(tempdir()),
  add.layer = FALSE,
  verbose = FALSE,
  get.map = TRUE,
  ...
)
```

Arguments

`x`	a `records` or an `rtoi` object.
`n`	the date expressed as the temporal index in the time series.
`...`	additional arguments
`lpos`	vector argument. Defines the position of the red-green-blue layers to enable a false color visualization.
`add.layer`	logical argument. If `TRUE`, the function plots the image on an existing map.
`verbose`	logical argument. If `TRUE`, the function prints the running steps and warnings.
`tmp_dir`	character argument. The directory where preview images are located.
`get.map`	logical argument. If `TRUE`, the function return the leaflet map.

Value

nothing. Previews the region in the viewer.

Examples

```
## Not run:
library(rsat)

# load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

set_credentials("username", "password")
set_database(file.path(tempdir(), "DATABASE"))

# by default the first date in rtoi is previewed
rsat_preview(navarre)


preview.dates <- dates(navarre)
# use add.layer to preview images of several days
rsat_preview(navarre,preview.dates[2],add.layer = TRUE)

# you can also preview records
rcrds <- records(navarre)
rsat_preview(rcrds, n = 1)

## End(Not run)
```

---

rsat_products *Show the products accepted by the services*

Description

Show the products accepted by the services

Usage

```
rsat_products(...)

## S4 method for signature 'ANY'
rsat_products()
```

Arguments

...	additional arguments.

Value

prints a list of products

Examples

```
rsat_products()
```

---

rsat_search *Search satellite images*

Description

Search satellite images concerning a particular location, data product, and date interval. The function returns a `records` object if the `region` is a `sf`. If an `rtoi` is used, the function returns nothing and the records are added to the `rtoi`.

Usage

```
rsat_search(region, product, ...)

## S4 method for signature 'rtoi,character'
rsat_search(region, product, verbose = FALSE, ...)

## S4 method for signature 'sf,character'
rsat_search(region, product, verbose = FALSE, ...)
```

Arguments

`region`	a `Spatial*`, `Raster*`, `sf` or `rtoi` class objects defining the region of interest.
`product`	a character vector of product names.
`...`	additional arguments for searching
`verbose`	logical argument. If `TRUE`, the function prints the running steps and warnings.

Details

MODIS images are found through the NASA Common Metadata Repository (CMR). The inventory of MODIS products can be found here. The catalog shows the product short names and detailed information. MODIS surface reflectance products are named 'mod09ga' and 'myd09ga' for Terra and Aqua satellites. By the time `rsat` is released, NASA carries out the maintenance of its website on Wednesdays, which may cause an error when connecting to their server.

We use ESA's powered API ('SciHub') to find Sentinel images. The catalog of Sentinel-2 and -3 products can be found here and here, respectively. Sentinel-2 and -3 surface reflectance product names are referred to as 'S2MSI2A' and 'SY_2_SYN___'.

Landsat images are accessed via the Machine-to-Machine API. Details about the Landsat products can be found here. The names of Landsat products are 'LANDSAT_TM_C1', 'LANDSAT_ETM_C1', and 'LANDSAT_8_C1' for missions 4-5, 7, and 8.

Value

nothing if x is an rtoi, records class if you search a region.

Examples

```
## Not run:
library(rsat)
set_credentials("username", "password")

# search navarre images using sf
record.list <- rsat_search(
  region = ex.navarre,
  product = "mod09ga",
  dates = as.Date("2011-01-01") + seq(1, 10, 1)
)

# creating a new rtoi
rtoi.path <- tempdir()
navarre <- new_rtoi(
  "Navarre", # name of the region
  ex.navarre, # sf of the region
  rtoi.path
) # path for the rtoi

# see the number of records in navarre
print(navarre)

# search modis images using rtoi
rsat_search(
  region = navarre,
  product = "mod09ga",
  dates = as.Date("2011-01-01") + seq(1, 10, 1)
)

# see the number of records in navarre
print(navarre)

# search landsat images using rtoi
rsat_search(
  region = navarre,
  product = "LANDSAT_8_C1",
  dates = as.Date("2016-01-01") + seq(1, 30, 1)
)

# see the number of records in navarre
print(navarre)

# search sentinel-2 (level 1 and level 2) images using rtoi
rsat_search(
  region = navarre,
  product = c("S2MSI1C", "S2MSI2A"),
  dates = as.Date("2016-01-01") + seq(1, 30, 1)
```

```
)

# see the number of records in navarre
print(navarre)

# search sentinel-3 level-2 images using rtoi
rsat_search(
  region = navarre,
  product = "OL_2_LFR___",
  dates = as.Date("2019-01-01") + seq(1, 2, 1)
)

# search sentinel-1 level-2 images using rtoi
rsat_search(
  region = navarre,
  product = "GRD",
  dates = as.Date("2019-01-01") + seq(1, 2, 1)
)

# search Landsat-5 images using rtoi
rsat_search(
  region = navarre,
  product = "LANDSAT_TM_C1",
  dates = as.Date("1988-08-01") + seq(1, 35)
)

print(navarre)

# get all records from rtoi
navarre.records <- records(navarre)

print(navarre.records)

## End(Not run)
```

`rsat_smoothing_images` *Fill data gaps and smooth outliers in a time series of satellite images*

Description

`apply_ima` is the implementation of a spatio-temporal method called Interpolation of Mean Anomalies(IMA) for gap filling and smoothing satellite data (Militino et al. 2019). `smoothing_images` is the implementation of a spatio temporal method called image mean anomaly (IMA) for gap filling and smoothing satellite data (Militino et al. 2019).

Usage

```
rsat_smoothing_images(x, method, ...)

## S4 method for signature 'rtoi,character'
```

```
rsat_smoothing_images(
  x,
  method,
  product = "ALL",
  satellite = "ALL",
  stage = "ALL",
  variable = "ALL",
  test.mode = FALSE,
  ...
)

## S4 method for signature 'SpatRaster,character'
rsat_smoothing_images(x, method, ...)
```

Arguments

`x` — `rtoi` or `RastespatRaster` containing a time series of satellite images.

`method` — character argument. Defines the method used for processing the images, e.a. "IMA".

`...` — arguments for nested functions:

- `Img2Fill` a `vector` defining the images to be filled/smoothed.
- `r.dates` a `vector` of dates for the layers in `x`. Mandatory when layer names of `x` do not contain their capturing dates "`YYYYJJJ`" format.
- `nDays` a `numeric` argument with the number of previous and subsequent days of the temporal neighborhood.
- `nYears` a `numeric` argument with the number of previous and subsequent years of the temporal neighborhood.
- `aFilter` a `vector` of lower and upper quantiles defining the outliers in the anomalies. Ex. c(0.05,0.95).
- `fact` a `numeric` argument specifying the aggregation factor of the anomalies.
- `fun` a `function` used to aggregate the image of anomalies. Both `mean` (default) or `median` are accepted.
- `snow.mode` logical argument. If `TRUE`, the process is parallelized using the functionalities from the ‘ `raster`’ package.
- `predictSE` calculate the standard error instead the prediction.
- `factSE` the `fact` used in the standard error prediction.
- `out.name` the name of the folder containing the smoothed/filled images when saved in the Hard Disk Device (HDD).
- `only.na` logical argument. If `TRUE` only fills the `NA` values. `FALSE` by default.

`product` — character argument. The name of the product to be processed. Check the name of the parameter with `rsat_list_data` function. Check the name of the parameter with `rsat_list_data` function. By default, "ALL".

`satellite` — character argument. The name of the satellite to be processed. Check the name of the parameter with `rsat_list_data` function. By default, "ALL".

stage	character argument. The name of the processed stage of the data. Check the name of the parameter with rsat_list_data function. By default, "ALL".
variable	character argument.The name of the variable to be processed. Check the name of the parameter with rsat_list_data function. By default, "ALL".
test.mode	logical argument. If TRUE, the function runs some lines to test rsat_smoothing_images with rtoi object.

Details

This filling/smoothing method was developed by Militino et al. (2019). IMA fills the gaps borrowing information from an adaptable temporal neighborhood. Two parameters determine the size of the neighborhood; the number of days before and after the target image (nDays) and the number of previous and subsequent years (nYears). Both parameters should be adjusted based on the temporal resolution of the of the time-series of images. We recommend that the neighborhood extends over days rather than years, when there is little resemblance between seasons. Also, cloudy series may require larger neighborhoods.

IMA gives the following steps; (1) creates a representative image from the temporal neighborhood of the target image (image to be filled/smoothed) e.g., doing the mean, median, etc. for each pixel's time-series (fun), (2) the target and representative images are subtracted giving an image of anomalies, (3) the anomalies falling outside the quantile limits (aFilter) are considered outliers and therefore removed, (4) it aggregates the anomaly image into a coarser resolution (fact) to reveal potential spatial dependencies, (5) the procedure fits a spatial model (thin plate splines or TPS) to the anomalies which is then used to interpolate the values at the original resolution, and (6) the output is the sum of the interpolated anomalies and the average image.

Value

a RastespatRaster with the filled/smoothed images.

References

Militino AF, Ugarte MD, Perez-Goya U, Genton MG (2019). "Interpolation of the Mean Anomalies for Cloud-Filling in Land Surface Temperature (LST) and Normalized Difference Vegetation Index (NDVI)." *IEEE Transactions on Geoscience and Remote Sensing. (Open-Access).* http://dx.doi.org/10.1109/TGRS.2019.2904193.

Examples

```
## Not run:
## Smooth data in rtoi
library(rsat)
require(terra)

# create a copy of pamplona in temp file
file.copy(from=system.file("ex/PamplonaDerived",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

# load example rtoi
```

```
pamplona <- read_rtoi(file.path(tempdir(),"PamplonaDerived"))
rsat_smoothing_images(pamplona,
                      method = "IMA",
                      variable="NDVI"
)
rsat_list_data(pamplona)
# get smoothed
smoothed <- rsat_get_SpatRaster(pamplona,p="mod09ga",v="NDVI",s="IMA")
plot(smoothed)

# get original
original <- rsat_get_SpatRaster(pamplona,p="mod09ga",v="NDVI",s="variables")
plot(original)
plot(smoothed[[1]]-original[[1]])

## smooth user defined SpatRaster dataset
require(terra)
data(ex.ndvi.navarre)

# load an example of NDVI time series in Navarre
ex.ndvi.navarre <- rast(ex.ndvi.navarre)
# the raster stack with the date in julian format as name
plot(ex.ndvi.navarre)

# smoothin and fill all the time series
tiles.mod.ndvi.filled <- rsat_smoothing_images(ex.ndvi.navarre,
  method = "IMA"
)
# show the filled images
plot(tiles.mod.ndvi.filled)

# plot comparison of the cloud and the filled images
tiles.mod.ndvi.comp <- c(
  ex.ndvi.navarre[[1]], tiles.mod.ndvi.filled[[1]],
  ex.ndvi.navarre[[2]], tiles.mod.ndvi.filled[[2]]
)
plot(tiles.mod.ndvi.comp)

## End(Not run)
```

`rtoi-class` *Region and Time Of Interest (*`rtoi`*)*

Description

It is a proxy object to store metadata about satellite imagery covering a spatial region over a time period. Images can come from multiple missions/programs and its purpose is to help managing heterogeneous datasets.

Details

An `rtoi` object manages two main folders called database and rtoi. The database is meant to work as a local, generic, and organized archive of raw satellite data retrieved with the `download()` function. The rtoi folder contains processed information for a particular region and time of interest. When `mosaic()` is called, the function crops and mosaics the relevant raw images from the database and saves the results in the rtoi folder. This folder also contains a `region.rtoi` file which saves metadata about the region/time of interest and satellite imagery available.

Fields

`name` a character with the name of the region of interest.

`rtoi_path` a character with the path to the rtoi folder.

`region` an sf with the region of interest.

`records` the satellite records available for your region and time of interest.

`db_path` a character with the path to the database.

Examples

```
## Not run:
data(ex.navarre)
## Create an rtoi with database
# path where the region is stored
rtoi.path <- tempdir()

# path where downloads are stored
db.path <- file.path(tempdir(), "DATABASE")
navarre <- new_rtoi(
  name = "Navarre_rtoi",
  region = ex.navarre,
  rtoi_path = rtoi.path,
  db_path = db.path
)

print(navarre)

## Create an rtoi without database
navarre2 <- new_rtoi(
  name = "Navarre_rtoi2",
  region = ex.navarre,
  rtoi_path = rtoi.path
)

print(navarre2)

## End(Not run)
```

sat_name *Get the name of the satellite(s) from a* `records` *or an* `rtoi`

Description

Get the name of the satellite(s) from a `records` or an `rtoi`

Usage

```
sat_name(x)

## S4 method for signature 'records'
sat_name(x)

## S4 method for signature 'rtoi'
sat_name(x)
```

Arguments

x	a `records` or an `rtoi` object.

Value

the name of the satellite

Examples

```
## Not run:
# load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

navarre <- read_rtoi(file.path(tempdir(),"Navarre"))
# get the records
rcds <- records(navarre)
# coerce the records to dataframe
sat_name(rcds)

## End(Not run)
```

set_credentials *Saves the credentials for the web services*

Description

Saves the credentials for the web services

Usage

```
set_credentials(user, pass, credential)

## S4 method for signature 'character,character,missing'
set_credentials(user, pass)

## S4 method for signature 'character,character,character'
set_credentials(user, pass, credential)
```

Arguments

user	character argument. Defines the username of an api platform to search or download images
pass	character argument. Defines the password of an api platform to search and download images
credential	optional argument to specify the name of the platform. Valid names are earthdata, scihub, scihubs5p, or ALL

Value

nothing. set the credentials in the package environment variable

Examples

```
print_credentials()
set_credentials("example", "example")
print_credentials()
set_credentials("example", "example", "earthdata")
print_credentials()
```

show,extent_crs-method

Show an Object

Description

Display the object, by printing, plotting or whatever suits its class. This function exists to be specialized by methods. The default method calls showDefault.

Usage

```
## S4 method for signature 'extent_crs'
show(object)

## S4 method for signature 'records'
show(object)

## S4 method for signature 'rtoi'
show(object)

## S4 method for signature 'variables'
show(object)
```

Arguments

object	Any R object

Value

show returns an invisible NULL.

Examples

```
## Not run:
## load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

## The method will now be used for automatic printing of navarre
navarre

## get records
rcds <- records(navarre)

rcds
```

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

`show_variables` *List the variables and satellites supported by* `rsat`

Description

Displays the satellites and variable method

Usage

```
show_variables(...)

## S4 method for signature 'ANY'
show_variables()
```

Arguments

`...`	arguments for nesting functions

Value

prints supported satellites and derived variables information.

Examples

```
show_variables()
```

`subset,records-method` *Filter the satellite records of a* `records` *or an* `rtoi`

Description

Filter the satellite records of a `records` or an `rtoi`

Usage

```
## S4 method for signature 'records'
subset(x, subset, select)
```

Arguments

`x`	a `records` or an `rtoi` object.
`subset`	character argument indicating the name of the slot.
`select`	character with the value for subsetting.

Value

filtered records class

test_function *Testing function*

Description

Function used for testing some internal functions in continuous integration.

Usage

```
test_function()
```

Examples

```
test_function()
```

tiles.mod.ndvi.filled.1.res *Result of IMA test 1*

Description

Filled image to test the package

tiles.mod.ndvi.filled.2.res *Result of IMA test 2*

Description

Filled image to test the package

unique,records,ANY-method

Extract unique elements

Description

It returns a `records` like `x` but with duplicate elements/rows removed.

Usage

```
## S4 method for signature 'records,ANY'
unique(x)
```

Arguments

`x`	a `records` object.

Value

unique elements in records class

Examples

```
## Not run:
# load example rtoi
file.copy(from=system.file("ex/Navarre",package="rsat"),
         to=tempdir(),
         recursive = TRUE)

navarre <- read_rtoi(file.path(tempdir(),"Navarre"))

# get the records
rcds <- records(navarre)

duplicate.records <- c(rcds[1],rcds[1])
length(duplicate.records)
print(duplicate.records)
single.record <- unique(duplicate.records)
length(single.record)
print(single.record)

## End(Not run)
```

[,extent_crs,ANY,ANY,ANY-method

Extract or replace parts of an object

Description

Operators acting on vectors, matrices, arrays and lists to extract or replace parts.

Usage

```
## S4 method for signature 'extent_crs,ANY,ANY,ANY'
x[i]

## S4 replacement method for signature 'extent_crs,ANY,ANY,ANY'
x[i] <- value

## S4 method for signature 'records,ANY,ANY,ANY'
x[i]

## S4 replacement method for signature 'records,ANY,ANY,ANY'
x[i] <- value
```

Arguments

x	object from which to extract element(s) or in which to replace element(s).
i	numeric argument. The the position of the element to select/modify.
value	a records argument. The slot of the records to be changed.

Value

returns a selected value

Index