

# Package ‘letsR’

August 29, 2016

**Type** Package

**Title** Tools for Data Handling and Analysis in Macroecology

**Version** 2.6

**Date** 2015-06-1

**Author** Bruno Vilela & Fabricio Villalobos

**Maintainer** Bruno Vilela <brunovilelasilva@hotmail.com>

**Description** R functions for handling, processing, and analyzing geographic data on species' distributions and environmental variables as well as species' description and conservation information.

**License** GPL-2

**Imports** XML, geosphere, fields, maptools, sp, rgdal, rgeos, methods

**Depends** R (>= 3.1.0), raster, maps

**Suggests** testthat

**LazyData** true

**URL** <http://onlinelibrary.wiley.com/doi/10.1111/2041-210X.12401/abstract>,  
<https://github.com/macroecology/letsR>

**BugReports** <https://github.com/macroecology/letsR/issues>

**RoxygenNote** 5.0.1

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2016-06-02 18:16:05

## R topics documented:

letsR-package . . . . .	2
IUCN . . . . .	3
lets.addpoly . . . . .	4
lets.addvar . . . . .	5
lets.classvar . . . . .	6

lets.correl . . . . .	7
lets.dismat . . . . .	8
lets.field . . . . .	9
lets.gridirizer . . . . .	10
lets.iucn . . . . .	11
lets.iucn.ha . . . . .	12
lets.iucn.his . . . . .	13
lets.iucncont . . . . .	15
lets.maplizer . . . . .	16
lets.midpoint . . . . .	17
lets.overlap . . . . .	18
lets.pamcrop . . . . .	19
lets.presab . . . . .	20
lets.presab.birds . . . . .	22
lets.presab.grid . . . . .	24
lets.presab.points . . . . .	26
lets.rangesize . . . . .	28
lets.shFilter . . . . .	29
lets.subsetPAM . . . . .	30
lets.summarizer . . . . .	31
lets.transf . . . . .	32
PAM . . . . .	33
Phyllomedusa . . . . .	34
plot.PresenceAbsence . . . . .	34
PresenceAbsence . . . . .	35
print.PresenceAbsence . . . . .	36
print.summary.PresenceAbsence . . . . .	37
summary.PresenceAbsence . . . . .	37
temp . . . . .	38
<b>Index</b>	<b>39</b>

## Description

The letsR package is being developed to help researchers in the handling, processing, and analysis of macroecological data. Its purpose is to integrate these methodological processes into a single software platform for macroecological analyses. The package's main functions allow users to build presence-absence matrices, the basic analytical tool in macroecology, from species' geographical distributions and merge them with species' traits, conservation information (downloadable using functions from this package) and spatial environmental layers. In addition, other package's functions enable users to summarize and visualize information from presence-absence matrices.

## Details

All functions in this package use a prefix and a suffix separated by a dot. The prefix refers to the package's name and the suffix to the actual function. This is done to avoid confusion with potentially similarly-named functions from other R packages. For instance, the `letsR` function used to create presence-absence matrices is called `lets.presab` (but see also `lets.presab.birds` and `lets.presab.points`) whereas the one used to add variables to a presence-absence matrix is called `lets.addvar`. The package's basic functions create and work on a particular S3 object class called `PresenceAbsence`. Such `PresenceAbsence` object class allows storing information beyond presence-absence data (e.g. user-defined grid-cell system) and using the generic `plot`, `summary` and `print` functions of R. Also, some package's functions allow the user to input customary R objects (e.g. vector, matrix, data.frame).

Another set of functions in this package allow the user to download species' information related to their description and conservation status as provided by the IUCN's RedList database (`lets.iucn`, `lets.iucn.ha`, `lets.iucn.his`). For this, such functions use the IUCN's RedList API to retrieve information from its webpage.

If you are looking for the most recent version of the package, you can get the development version of `letsR` on github (<http://www.github.com/macroecology/letsR>).

Package: lestR  
Type: Package  
Version: 2.6  
Date: 2015-06-01  
License: GPL-2

## Author(s)

### Bruno Vilela

(email: <[bvilela@wustl.edu](mailto:bvilela@wustl.edu)>; Website: <http://bvilela.weebly.com/>)

### Fabricio Villalobos

(email: <[fabricio.villalobos@gmail.com](mailto:fabricio.villalobos@gmail.com)>; Website: <https://sites.google.com/site/fabriciovillalobos/>)

## References

Vilela, B., & Villalobos, F. (2015). `letsR`: a new R package for data handling and analysis in macroecology. *Methods in Ecology and Evolution*.

---

IUCN

*IUCN avaliation for frogs of Phyllomedusa genus*

---

## Description

Result of the function `lets.iucn` applied to Southern American frog genera *Phyllomedusa* in 2014.

## Usage

```
data(IUCN)
```

lets.addpoly

*Add polygon coverage to a PresenceAbsence object*

---

**Description**

Add polygon coverage within cells of a PresenceAbsence object.

**Usage**

```
lets.addpoly(x, y, z, onlyvar = FALSE)
```

**Arguments**

x	A <a href="#">PresenceAbsence</a> object.
y	Polygon of interest.
z	A character indicating the column name of the polygon containing the attributes to be used.
onlyvar	If TRUE only the matrix object will be returned.

**Value**

The result is a presence-absence matrix of species with the polygons' attributes used added as columns at the right-end of the matrix . The Values represent the percentage of the cell covered by the polygon attribute used.

**Author(s)**

Bruno Vilela

**See Also**

[lets.presab.birds](#)  
[lets.presab](#)  
[lets.addvar](#)

**Examples**

```
## Not run:  
data(PAM) # Phyllomedusa presence-absence matrix  
require(maptools)  
data(wrld_simpl) # World map  
Brazil <- wrld_simpl[wrld_simpl$NAME == "Brazil", ] # Brazil (polygon)  
  
# Check where is the variable name  
# (in this case it is in "NAME" which will be my z value)  
names(Brazil)
```

```
PAM_pol <- lets.addpoly(PAM, Brazil, "NAME")  
## End(Not run)
```

---

`lets.addvar`*Add variables (in raster format) to a PresenceAbsence object*

---

### Description

Add variables (in raster format), usually environmental, to a PresenceAbsence object. Variables are included as additional columns containing the aggregate/summarize value of the variable(s) in each cell of the presence-absence matrix.

### Usage

```
lets.addvar(x, y, onlyvar = FALSE, fun = mean)
```

### Arguments

<code>x</code>	A <a href="#">PresenceAbsence</a> object.
<code>y</code>	Variables to be added in Raster or RasterStack format.
<code>onlyvar</code>	If TRUE only the matrix object will be returned.
<code>fun</code>	Function used to aggregate the variables(s) values over each cell. Note that this will only work for variables with a resolution value smaller (i.e. higher resolution) than the PAM.

### Value

The result is a presence-absence matrix of species with the variables added as columns at the right-end of the matrix (but see the 'onlyvar' argument).

### Note

The PresenceAbsence and the Raster variable must be in the same projection.

### Author(s)

Bruno Vilela

### See Also

[lets.presab.birds](#)

[lets.presab](#)

[lets.addpoly](#)

**Examples**

```
## Not run:
data(temp) # Global mean temperature
data(PAM) # Phyllomedusa presence-absence matrix
# Mean temperature
PAM_temp_mean <- lets.addvar(PAM, temp)
# Standard deviation of temperature
PAM_temp_sd <- lets.addvar(PAM, temp, fun = sd, onlyvar = TRUE)
# Mean and SD in the PAM
PAM_temp_mean_sd <- cbind(PAM_temp_mean, PAM_temp_sd)

## End(Not run)
```

lets.classvar

*Frequency distribution of a variable within a species' range***Description**

Based on a species Presence-Absence matrix including variables of interest (see [lets.addvar](#)), the function divides a continuous variable into classes and counts the frequency of each class within each species' range.

**Usage**

```
lets.classvar(x, groups = "default", pos, xy)
```

**Arguments**

x	Presence-absence matrix with a single variable added (see <a href="#">lets.addvar</a> ).
groups	The number of classes into which the variable will be divided. Default calculates the number of classes as the default for a histogram ( <a href="#">hist</a> ).
pos	Column number containing the variable of interest.
xy	Logical, if TRUE the input matrix contains the geographic coordinates in the first two columns.

**Value**

A matrix with species in the rows and the variable's classes in the columns.

**Author(s)**

Bruno Vilela

**References**

Morales-Castilla et al. 2013. Range size patterns of New World oscine passerines (Aves): insights from differences among migratory and sedentary clades. *Journal of Biogeography*, 40, 2261-2273.

**Examples**

```
## Not run:
data(PAM)
data(temp)
pamvar <- lets.addvar(PAM, temp)
resu <- lets.classvar(x = pamvar, pos = ncol(pamvar), xy = TRUE)

## End(Not run)
```

lets.correl

*Compute correlogram based on the Moran's I index***Description**

Computes the Moran's I correlogram of a single or multiple variables.

**Usage**

```
lets.correl(x, y, z, equidistant = FALSE, plot = TRUE)
```

**Arguments**

x	A single numeric variable in vector format or multiple variables in matrix format (as columns).
y	A distance matrix of class <code>matrix</code> or <code>dist</code> .
z	The number of distance classes to use in the correlogram.
equidistant	Logical, if TRUE the classes will be equidistant. If FALSE the classes will have equal number of observations.
plot	Logical, if TRUE the correlogram will be plotted.

**Value**

Returns a matrix with the Moran's I Observed value, Confidence Interval (95 and Expected value. Also the p value of the randomization test, the mean distance between classes, and the number of observations. quase tudo

**Author(s)**

Bruno Vilela, Fabricio Villalobos, Lucas Jardim & Jose Alexandre Diniz-Filho

**References**

Sokal, R.R. & Oden, N.L. (1978) Spatial autocorrelation in biology. 1. Methodology. *Biological Journal of the Linnean Society*, 10, 199-228.

Sokal, R.R. & Oden, N.L. (1978) Spatial autocorrelation in biology. 2. Some biological implications and four applications of evolutionary and ecological interest. *Biological Journal of the Linnean Society*, 10, 229-249.

## Examples

```
## Not run:
data(PAM)
data(IUCN)

# Spatial autocorrelation in description year (species level)
midpoint <- lets.midpoint(PAM)
distan <- lets.distmat(midpoint[, 2:3])
moran <- lets.correl(IUCN$Description, distan, 12,
                    equidistant = FALSE,
                    plot = TRUE)

## End(Not run)
```

---

lets.distmat	<i>Compute a geographic distance matrix</i>
--------------	---

---

## Description

Calculates a geographic distance matrix based on a PresenceAbsence or a two column matrix of x(longitude) and y(latitude).

## Usage

```
lets.distmat(xy, asdist = TRUE, ...)
```

## Arguments

xy	A <a href="#">PresenceAbsence</a> object or a matrix with two columns (longitude, latitude).
asdist	Logical, if TRUE the result will be an object of class <code>dist</code> , if FALSE the result will be an object of class <code>matrix</code> .
...	Arguments to be passed to the function <code>rdist.earth</code> of package <code>fields</code> .

## Details

This function basically facilitates the use of `rdist.earth` on a PresenceAbsence object, allowing also the user to have directly a `dist` object.

## Value

The user can choose between `dist` and `matrix` class object to be returned. The resulting values are in kilometers (but see the argument 'miles' in `rdist.earth`).

## Author(s)

Bruno Vilela & Fabricio Villalobos



## Examples

```
## Not run:  
data(PAM)  
distPAM <- lets.distmat(PAM)  
  
## End(Not run)
```

---

lets.field	<i>Create species' values based on the species co-occurrence within focal ranges</i>
------------	--

---

## Description

Create single species' values based on the attributes of species co-occurring within individual ranges.

## Usage

```
lets.field(x, y, z, weight = TRUE, xy = NULL, count = FALSE)
```

## Arguments

x	A <a href="#">PresenceAbsence</a> object or a presence-absence in matrix format (see xy argument for matrix use) with the species named in the columns.
y	Species attribute to be considered. It must be a numeric attribute.
z	Species names in the same order as the attributes and exactly the same as named in the matrix or in the PresenceAbsence object.
weight	If TRUE the value is weighted by species' range size, if FALSE the value is the mean of all species that co-occur within the focal species.
xy	If TRUE the presence-absence matrix contains the cell coordinates in the first two columns.
count	Logical, if TRUE a counting window will open.

## Details

If the species do not co-occur with any other species NaN will be returned.

## Author(s)

Bruno Vilela & Fabricio Villalobos

## References

Villalobos, F. and Arita, H.T. 2010. The diversity field of New World leaf-nosed bats (Phyllostomidae). *Global Ecology and Biogeography*. 19, 200-211.

Villalobos, F., Rangel, T.F., and Diniz-Filho, J.A.F. 2013. Phylogenetic fields of species: cross-species patterns of phylogenetic structure and geographical coexistence. *Proceedings of the Royal Society B*. 280, 20122570.

## See Also

[lets.presab.birds](#)

[lets.presab](#)

## Examples

```
## Not run:
data(PAM)
range <- lets.rangesize(x = PAM, units = "cell")
field <- lets.field(PAM, range, PAM$S, weight = TRUE)

## End(Not run)
```

---

lets.gridirizer	<i>Fits a grid into a PresenceAbsence object</i>
-----------------	--

---

## Description

This function creates a grid in shapefile format and adds its cells' IDs to the presence-absence matrix. The function was created to facilitate the use of the PresenceAbsence object for the ones who prefer to work with a grid in shapefile format.

## Usage

```
lets.gridirizer(x)
```

## Arguments

x                    A [PresenceAbsence](#) object.

## Value

The result is a list of two objects. The first is a grid in shapefile format; the second is a presence-absence matrix with an additional column called SP\_ID (shapefile cell identifier).

## Author(s)

Bruno Vilela

**See Also**[plot.PresenceAbsence](#)[lets.presab.birds](#)**Examples**

```
## Not run:
data(PAM)
PAM.grid <- lets.gridirizer(PAM)
names(PAM.grid)
# Grid in polygon format (can be saved in shapefile)
PAM.grid$Grid
# Presence-absence matrix (beginning only)
head(PAM.grid$Presence[, 1:5])

## End(Not run)
```

---

lets.iucn	<i>Download species' information from the IUCN RedList online database</i>
-----------	--

---

**Description**

Get species' information from the IUCN website(<http://www.iucnredlist.org/>) for one or more species.

**Usage**

```
lets.iucn(input, count = FALSE)
```

**Arguments**

input	Character vector with one or more species names, or an object of class <a href="#">PresenceAbsence</a> .
count	Logical, if TRUE a counting window will open.

**Details**

Note that you must be connected to the internet to use this function.

**Value**

Returns a data frame with the Species Name, Family, Conservation Status, Criteria used to establish the conservation status, Population Status, Year of Description (only for animals), and the Countries where it occurs. If species do not have information (i.e. have not been evaluated), the result is: NE (Not evaluated).

**Author(s)**

Bruno Vilela

**See Also**[lets.iucn.ha](#)[lets.iucn.his](#)**Examples**

```
## Not run:
# Single species
lets.iucn("Pongo pygmaeus")

# Multiple species
sp <- c("Musonycteris harrisoni", "Ailuropoda melanoleuca",
       "Cebus flavius")
lets.iucn(sp)

## End(Not run)
```

lets.iucn.ha

---

*Download species' habitat information from the IUCN RedList online database*

---

**Description**

Get species' habitat information from the IUCN RedList website(<http://www.iucnredlist.org/>) for one or more species.

**Usage**

```
lets.iucn.ha(input, count = FALSE)
```

**Arguments**

input	Character vector with one or more species names, or an object of the <a href="#">PresenceAbsence</a> class.
count	Logical, if TRUE a counting window will open.

**Details**

Note that you must be connected to the internet to use this function.

**Value**

A data frame with species names in the first column and the habitats where it occurs in the remaining columns, '1' if species is present in that habitat and '0' otherwise.

**Author(s)**

Bruno Vilela

**See Also**[lets.iucn](#)[lets.iucn.his](#)**Examples**

```
## Not run:
# Single species
lets.iucn.ha("Pongo pygmaeus")

# Multiple species
sp <- c("Musonycteris harrisoni", "Ailuropoda melanoleuca",
       "Cebus flavius")
lets.iucn.ha(sp)

## End(Not run)
```

---

lets.iucn.his	<i>Download species' temporal trend in conservation status from the IUCN RedList online database</i>
---------------	--

---

**Description**

Get species conservation status over time (i.e. from 1980 to the present date available) from the IUCN website(<http://www.iucnredlist.org/>) for one or more species.

**Usage**

```
lets.iucn.his(input, count = FALSE)
```

**Arguments**

input	character vector with one or more species names, or an object of class <a href="#">PresenceAbsence</a> .
count	Logical, if TRUE a counting window will open.

**Details**

Note that you must be connected to the internet to use this function.

**Value**

A data frame with the species names in the first column rows and the years (1980 - present) in the remaining columns, the code represents the species' conservation status (see the IUCN RedList website for details). If species do not have information (i.e. have not been evaluated), the result is: NE (Not evaluated).

Codes and categories:

**EX:** Extinct

**EW:** Extinct in the Wild

**VU:** Vulnerable

**EN:** Endangered

**CR:** Critically Endangered

**LC:** Least Concern

**NT:** Near Threatened

**DD:** Data Deficient

**CT:** Commercially Threatened

**IN:** Indeterminate

**IK:** Insufficiently Known

**LR:** Lower Risk

**RA:** Rare

**Author(s)**

Bruno Vilela

**See Also**

[lets.iucn.ha](#)

[lets.iucn](#)

**Examples**

```
## Not run:
# Single species
lets.iucn.his("Panthera onca")

# Multiple species
sp <- c("Rhincodon typus", "Ailuropoda melanoleuca")
lets.iucn.his(sp)

## End(Not run)
```

---

lets.iucncont                      *Transform IUCN RedList conservation status to continuous values*

---

**Description**

Transform IUCN RedList conservation status to continuous values ranging from 0 to 5.

**Usage**

```
lets.iucncont(x, dd = NA, ne = NA)
```

**Arguments**

x	A vector or a matrix containing IUCN codes to be transformed.
dd	The value to be attributed to DD (data-deficient) species, the default option is NA.
ne	The value to be attributed to NE (not-evaluated) species, the default option is NA.

**Value**

Returns a vector/matrix with continuous values from 0 to 5.

EX and EW = 5

CR = 4

EN = 3

VU = 2

NT = 1

LC = 0

DD = NA

NE = NA

**Author(s)**

Bruno Vilela

**References**

Purvis A et al., 2000. Predicting extinction risk in declining species. Proceedings of the Royal Society of London. Series B: Biological Sciences, 267.1456: 1947-1952.

**See Also**

[lets.iucn](#)

**Examples**

```
## Not run:
#Vector transformation
status <- sample(c("EN","VU", "NT", "CR", "DD", "LC", "EX"),
                30, replace = TRUE)
transV <- lets.iucncont(status)

#matrix transformation
data(IUCN)
transM <- lets.iucncont(IUCN)

## End(Not run)
```

---

lets.maplizer	<i>Create a matrix summarizing species' attributes within cells of a PresenceAbsence object</i>
---------------	---

---

**Description**

Summarize species attributes per cell in a presence-absence matrix.

**Usage**

```
lets.maplizer(x, y, z, func = mean, ras = FALSE)
```

**Arguments**

x	A <a href="#">PresenceAbsence</a> object.
y	Species attribute to be considered. It must be a numeric attribute.
z	Species names in the same order as the attributes and exactly the same as named in the PresenceAbsence object.
func	A function to summarize the species' attribute in each cell (the function must return only one value).
ras	If TRUE the raster object will be returned together with the matrix.

**Value**

The result can be both a matrix or a list containing the follow objects:

**Matrix:** a matrix object with the cells' geographic coordinates and the summarized species' attributes within them.

**Raster:** The summarized species' attributed mapped in a raster object.

**Author(s)**

Bruno Vilela



**See Also**[lets.presab](#)[lets.presab.birds](#)**Examples**

```
## Not run:
data(PAM)
data(IUCN)
trait <- IUCN$Description_Year
resu <- lets.maplizer(PAM, trait, PAM$$, ras = TRUE)
head(resu$Matrix)
plot(resu$Raster, xlab = "Longitude", ylab = "Latitude",
main = "Mean description year per site") ; map(add = TRUE)

## End(Not run)
```

---

`lets.midpoint`*Compute the midpoint of species' geographic ranges*

---

**Description**

Calculate species' distributional midpoint from a presence-absence matrix.

**Usage**

```
lets.midpoint(pam, planar = FALSE)
```

**Arguments**

<code>pam</code>	A presence-absence matrix (sites in the rows and species in the columns, with the first two columns containing the longitudinal and latitudinal coordinates, respectively), or an object of class <a href="#">PresenceAbsence</a> .
<code>planar</code>	Logical, if FALSE the coordinates are in Longitude/Latitude. If TRUE the coordinates are planar.

**Value**

A data.frame containing the species' names and geographic coordinates (longitude [x], latitude [y]) of species' midpoints.

**Author(s)**

Fabricio Villalobos & Bruno Vilela

**See Also**[lets.presab](#)[lets.presab.birds](#)**Examples**

```
## Not run:
data(PAM)
mid <- lets.midpoint(PAM)

## End(Not run)
```

lets.overlap

*Compute pairwise species' geographic overlaps***Description**

Creates a species geographic overlap matrix from a Presence-absence matrix.

**Usage**

```
lets.overlap(pam, method = "Chesser&Zink", xy = NULL)
```

**Arguments**

pam	A presence-absence matrix (sites in rows and species in columns, with the first two columns containing the longitudinal and latitudinal coordinates, respectively), or an object of class <a href="#">PresenceAbsence</a> .
method	The method used to calculate the overlap matrix. "Chesser&Zink" calculates the degree of overlap as the proportion of the smaller range that overlaps within the larger range (Chesser & Zink 1994). "Proportional" calculates the proportion of a range that overlaps another range, the resultant matrix is not symmetric. "Cells" will show the number of overlapping grid cells between a pair of species' ranges (same for both species in a pair), here the resultant matrix is symmetric.
xy	Logical, if TRUE the input matrix contains geographic coordinates in the first two columns.

**Author(s)**

Fabricio Villalobos & Bruno Vilela

**References**

Chesser, R. Terry, and Robert M. Zink. "Modes of speciation in birds: a test of Lynch's method." *Evolution* (1994): 490-497.

Barracough, Timothy G., and Alfred P. Vogler. "Detecting the geographical pattern of speciation from species-level phylogenies." *The American Naturalist* 155.4 (2000): 419-434.

**See Also**[lets.presab](#)[lets.presab.birds](#)**Examples**

```
## Not run:
data(PAM)
CZ <- lets.overlap(PAM, method = "Chesser&Zink")
prop <- lets.overlap(PAM, method = "Proportional")
cells <- lets.overlap(PAM, method = "Cells")

## End(Not run)
```

---

`lets.pamcrop`*Crop a PresenceAbsence object based on an input shapefile*

---

**Description**

Crop a PresenceAbsence object based on a shapefile provided by the user.

**Usage**

```
lets.pamcrop(x, shp, remove.sp = TRUE)
```

**Arguments**

<code>x</code>	A <a href="#">PresenceAbsence</a> object.
<code>shp</code>	Object of class <code>SpatialPolygonsDataFrame</code> (see function <a href="#">readShapePoly</a> ) to crop the PresenceAbsence object.
<code>remove.sp</code>	Logical, if TRUE the final matrix will not contain species that do not match any cell in the grid.

**Value**

The result is an object of class `PresenceAbsence` cropped.

**Author(s)**

Bruno Vilela

**See Also**[plot.PresenceAbsence](#)[lets.presab.birds](#)

## Examples

```
## Not run:
data(PAM)
# PAM before crop
plot(PAM, xlab = "Longitude", ylab = "Latitude",
     main = "Phyllomedusa species richness")

# Crop PAM to Brazil
require(maptools)
data(wrld_simpl) # World map
Brazil <- wrld_simpl[wrld_simpl$NAME == "Brazil", ] # Brazil (polygon)
PAM_crop <- lets.pamcrop(PAM, Brazil, remove.sp = TRUE)
plot(PAM_crop, xlab = "Longitude", ylab = "Latitude",
     main = "Phyllomedusa species richness (Brazil crop)",
     col = colorRampPalette(c("darkgreen", "yellow", "blue")))

## End(Not run)
```

---

lets.presab	<i>Create a presence-absence matrix of species' geographic ranges within a grid</i>
-------------	---

---

## Description

Convert species' ranges (in shapefile format) into a presence-absence matrix based on a user-defined grid system

## Usage

```
lets.presab(shapes, xmn = -180, xmx = 180, ymn = -90, ymx = 90,
            resol = 1, remove.cells = TRUE, remove.sp = TRUE, show.matrix = FALSE,
            crs = CRS("+proj=longlat +datum=WGS84"), crs.grid = crs, cover = 0,
            presence = NULL, origin = NULL, seasonal = NULL, count = FALSE)
```

## Arguments

shapes	Object of class <code>SpatialPolygonsDataFrame</code> (see function <a href="#">readShapePoly</a> to open these files) containing the distribution of one or more species. Species names should be in a column (within the <code>.DBF</code> table of the shapefile) called <code>BINOMIAL/binomial</code> or <code>SCINAME/sciname</code> .
xmn	Minimum longitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest)
xmx	Maximum longitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest)
ymn	Minimum latitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest)

<code>ymx</code>	Maximum latitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest)
<code>resol</code>	Numeric vector of length 1 or 2 to set the grid resolution.
<code>remove.cells</code>	Logical, if TRUE the final matrix will not contain cells in the grid with a value of zero (i.e. sites with no species present).
<code>remove.sp</code>	Logical, if TRUE the final matrix will not contain species that do not match any cell in the grid.
<code>show.matrix</code>	Logical, if TRUE only the presence-absence matrix will be returned.
<code>crs</code>	Character representign the PROJ.4 type description of a Coordinate Reference System (map projection) of the polygons.
<code>crs.grid</code>	Character representign the PROJ.4 type description of a Coordinate Reference System (map projection) for the grid. Note that when you change this options you may probably change the extent coordinates and the resolution.
<code>cover</code>	Percentage of the cell covered by the shapefile that will be considered for presence (values between 0 and 1). This option is only available when the coordinates are in degrees (longitude/latitude).
<code>presence</code>	A vector with the code numbers for the presence type to be considered in the process (for IUCN spatial data <a href="http://www.iucnredlist.org/technical-documents/spatial-data">http://www.iucnredlist.org/technical-documents/spatial-data</a> , see metadata).
<code>origin</code>	A vector with the code numbers for the origin type to be considered in the process (for IUCN spatial data).
<code>seasonal</code>	A vector with the code numbers for the seasonal type to be considered in the process (for IUCN spatial data).
<code>count</code>	Logical, if TRUE a counting window will open.

## Details

The function creates the presence-absence matrix based on a raster object. Depending on the cell size, extension used and number of species it may require a lot of memory, and may take some time to process it. Thus, during the process, if `count` argument is set TRUE, a counting window will open so you can see the progress (i.e. in what polygon/shapefile the function is working). Note that the number of polygons is not the same as the number of species that you have (i.e. a species may have more than one polygon/shapefiles).

## Value

The result is a list object of class `PresenceAbsence` with the following objects:

**Presence-Absence Matrix:** A matrix of species' presence(1) and absence(0) information. The first two columns contain the longitude (x) and latitude (y) of the cells' centroid (from the gridded domain used);

**Richness Raster:** A raster containing species richness data;

**Species name:** A character vector with species' names contained in the matrix.

\*But see the optional argument `show.matrix`.

**Author(s)**

Bruno Vilela & Fabricio Villalobos

**See Also**

[plot.PresenceAbsence](#)

[lets.presab.birds](#)

[lets.shFilter](#)

**Examples**

```
## Not run:
# Spatial distribution polygons of south american frogs
# of genus Phyllomedusa.
data(Phyllomedusa)
PAM <- lets.presab(Phyllomedusa, xmn = -93, xmx = -29,
                  ymn = -57, ymx = 15)
summary(PAM)
# Species richness map
plot(PAM, xlab = "Longitude", ylab = "Latitude",
     main = "Phyllomedusa species richness")
# Map of the specific species
plot(PAM, name = "Phyllomedusa nordestina")

## End(Not run)
```

---

lets.presab.birds	<i>Create a presence-absence matrix of species' geographic ranges within a grid for the Birdlife spatial data</i>
-------------------	---

---

**Description**

Convert species' ranges (in shapefile format and stored in particular folders) into a presence-absence matrix based on a user-defined grid. This function is specially designed to work with BirdLife Intl. shapefiles (<http://www.birdlife.org>).

**Usage**

```
lets.presab.birds(path, xmn = -180, xmx = 180, ymn = -90, ymx = 90,
                  resol = 1, remove.cells = TRUE, remove.sp = TRUE, show.matrix = FALSE,
                  crs = CRS("+proj=longlat +datum=WGS84"), crs.grid = crs, cover = 0,
                  presence = NULL, origin = NULL, seasonal = NULL, count = FALSE)
```

**Arguments**

path	Path location of folders with one or more species' individual shapefiles. Shapefiles with more than one species will not work on this function. To use multi-species shapefiles see <a href="#">lets.presab</a> .
xmn	Minimum longitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest).
xmx	Maximum longitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest).
ymn	Minimum latitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest).
ymx	Maximum latitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest).
resol	Numeric vector of length 1 or 2 to set the grid resolution.
remove.cells	Logical, if TRUE the final matrix will not contain cells in the grid with a value of zero (i.e. sites with no species present).
remove.sp	Logical, if TRUE the final matrix will not contain species that do not match any cell in the grid.
show.matrix	Logical, if TRUE only the presence-absence matrix will be returned.
crs	Character representing the PROJ.4 type description of a Coordinate Reference System (map projection) of the original polygons.
crs.grid	Character representing the PROJ.4 type description of a Coordinate Reference System (map projection) for the grid. Note that when you change this option you may probably change the extent coordinates and the resolution.
cover	Percentage of the cell covered by the shapefile that will be considered for presence (values between 0 and 1). This option is only available when the coordinates are in degrees (longitude/latitude).
presence	A vector with the code numbers for the presence type to be considered in the process (for IUCN spatial data <a href="http://www.iucnredlist.org/technical-documents/spatial-data">http://www.iucnredlist.org/technical-documents/spatial-data</a> , see metadata).
origin	A vector with the code numbers for the origin type to be considered in the process (for IUCN spatial data).
seasonal	A vector with the code numbers for the seasonal type to be considered in the process (for IUCN spatial data).
count	Logical, if TRUE a counting window will open.

**Details**

The function creates the presence-absence matrix based on a raster file. Depending on the cell size, extension used and number of species it may require a lot of memory, and may take some time to process it. Thus, during the process, if count argument is set TRUE, a counting window will open so you can see the progress (i.e. in what polygon the function is working). Note that the number of polygons is not the same as the number of species that you have (i.e. a species may have more than one polygon/shapefiles).

**Value**

The result is an object of class `PresenceAbsence` with the following objects:

**Presence-Absence Matrix:** A matrix of species' presence(1) and absence(0) information. The first two columns contain the longitude (x) and latitude (y) of the cells' centroid (from the gridded domain used);

**Richness Raster:** A raster containing species richness data;

**Species name:** A vector with species' names contained in the matrix.

\*But see the optional argument `show.matrix`.

**Author(s)**

Bruno Vilela & Fabricio Villalobos

**See Also**

[plot.PresenceAbsence](#)

[lets.presab](#)

[lets.shFilter](#)

**Examples**

```
## Not run:
# Constructing a Presence/Absence matrix for birds
# Attention: For your own files, omit the 'system.file'
# and 'package="letsR"', these are just to get the path
# to files installed with the package.
path.Ramphastos <- system.file("extdata", package = "letsR")
PAM <- lets.presab.birds(path.Ramphastos, xmn = -93, xmx = -29,
                        ymn = -57, ymx = 25)

# Species richness map
plot(PAM, xlab = "Longitude", ylab = "Latitude",
     main = "Ramphastos species Richness")

## End(Not run)
```

---

lets.presab.grid

*Create a presence-absence matrix of species' geographic ranges within a user's grid shapefile (beta version)*

---

**Description**

Convert species' ranges (in shapefile format) into a presence-absence matrix based on a grid in shapefile format.



**Usage**

```
lets.presab.grid(shapes, grid, sample.unit, remove.sp = TRUE,  
presence = NULL, origin = NULL, seasonal = NULL)
```

**Arguments**

shapes	Object of class SpatialPolygonsDataFrame (see function <a href="#">readShapePoly</a> to open these files) containing the distribution of one or more species. Species names should be in a column (within the .DBF table of the shapefile) called BINOMIAL/binomial or SCINAME/sciname.
grid	Object of class shapefile representing the spatial grid (e.g. regular/irregular cells, political divisions, hexagonal grids, etc). The grid and the shapefiles must be in the same projection.
sample.unit	Object of class character with the name of the column (in the grid) representing the sample units of the presence absence matrix.
remove.sp	Logical, if TRUE the final matrix will not contain species that do not match any cell in the grid.
presence	A vector with the code numbers for the presence type to be considered in the process (for IUCN spatial data <a href="http://www.iucnredlist.org/technical-documents/spatial-data">http://www.iucnredlist.org/technical-documents/spatial-data</a> , see metadata).
origin	A vector with the code numbers for the origin type to be considered in the process (for IUCN spatial data).
seasonal	A vector with the code numbers for the seasonal type to be considered in the process (for IUCN spatial data).

**Details**

This function is an alternative way to create a presence absence matrix when users already have their own grid.

**Value**

The result is a list containing two objects:

- (I) A matrix the species presence (1) and absence (0) values per sample unity.
- (II) The original grid.

**Author(s)**

Bruno Vilela & Fabricio Villalobos

**See Also**

[plot.PresenceAbsence](#)  
[lets.presab.birds](#)  
[lets.shFilter](#)

**Examples**

```
## Not run:
# Grid
sp.r <- rasterToPolygons(raster(resol = 5))
slot(sp.r, "data") <- cbind("ID" = 1:length(sp.r),
                           slot(sp.r, "data"))

# Species polygons
data(Phyllomedusa)
projection(Phyllomedusa) <- projection(sp.r)

# PAM
resu <- lets.presab.grid(Phyllomedusa, sp.r, "ID")

# Plot
rich_plus1 <- rowSums(resu$PAM) + 1
colfunc <- colorRampPalette(c("#fff5f0", "#fb6a4a", "#67000d"))
colors <- c("white", colfunc(max(rich_plus1)))
plot(resu$grid, border = "gray40",
      col = colors[rich_plus1])
map(add = TRUE)

## End(Not run)
```

---

lets.presab.points      *Create a presence-absence matrix based on species' point occurrences*

---

**Description**

Convert species' occurrences into a presence-absence matrix based on a user-defined grid.

**Usage**

```
lets.presab.points(xy, species, xmn = -180, xmx = 180, ymn = -90,
                  ymx = 90, resol = 1, remove.cells = TRUE, remove.sp = TRUE,
                  show.matrix = FALSE, crs = CRS("+proj=longlat +datum=WGS84"),
                  count = FALSE)
```

**Arguments**

xy	A matrix with geographic coordinates of species occurrences, first column is the longitude (or x), and the second latitude (or y).
species	Character vector with species names, in the same order as the coordinates.
xmn	Minimum longitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest)

xmx	Maximum longitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest)
ymn	Minimum latitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest)
ymx	Maximum latitude used to construct the grid in which the matrix will be based (i.e. the [gridded] geographic domain of interest)
resol	Numeric vector of length 1 or 2 to set the grid resolution.
remove.cells	Logical, if TRUE the final matrix will not contain cells in the grid with a value of zero (i.e. sites with no species present).
remove.sp	Logical, if TRUE the final matrix will not contain species that do not match any cell in the grid.
show.matrix	Logical, if TRUE only the presence-absence matrix will be returned.
crs	Character representing the PROJ.4 type description of a Coordinate Reference System (map projection) of the points.
count	Logical, if TRUE a counting window will open.

### Details

The function creates the presence-absence matrix based on a raster file. Depending on the cell size, extension used and number of species it may require a lot of memory, and may take some time to process it. Thus, during the process, if count argument is set TRUE, a counting window will open so you can see the progress (i.e. in what polygon the function is working). Note that the number of polygons is not the same as the number of species that you have (i.e. a species may have more than one polygon/shapefiles).

### Value

The result is an object of class [PresenceAbsence](#) with the following objects:

**Presence-Absence Matrix:** A matrix of species' presence(1) and absence(0) information. The first two columns contain the longitude (x) and latitude (y) of the cells' centroid (from the gridded domain used);

**Richness Raster:** A raster containing species richness data;

**Species name:** A character vector with species' names contained in the matrix.

\*But see the optional argument `show.matrix`.

### Author(s)

Bruno Vilela & Fabricio Villalobos

### See Also

[plot.PresenceAbsence](#)

[lets.presab.birds](#)

[lets.presab](#)

[lets.shFilter](#)

**Examples**

```
## Not run:
species <- c(rep("sp1", 100), rep("sp2", 100),
             rep("sp3", 100), rep("sp4", 100))
x <- runif(400, min = -69, max = -51)
y <- runif(400, min = -23, max = -4)
xy <- cbind(x, y)
PAM <- lets.presab.points(xy, species, xmn = -93, xmx = -29,
                         ymn = -57, ymx = 15)

summary(PAM)
# Species richness map
plot(PAM, xlab = "Longitude", ylab = "Latitude",
     main = "Species richness map (simulated)")

# Map of the specific species
plot(PAM, name = "sp1")

## End(Not run)
```

---

lets.rangesize

*Compute species' geographic range sizes*


---

**Description**

This function calculates species' range sizes from a PresenceAbsence object or directly from the species' shapefiles.

**Usage**

```
lets.rangesize(x, species_name = x@data[, 1], coordinates = "geographic",
              units = "cell")
```

**Arguments**

x	A <a href="#">PresenceAbsence</a> object or an <a href="#">SpatialPolygonsDataFrame</a> .
species_name	Species names in the same order as in the <a href="#">SpatialPolygonsDataFrame</a> (only needed if x is a <a href="#">SpatialPolygonsDataFrame</a> ).
coordinates	"geographical" or "planar". Indicate wheter the shapefile has geographical or planar coordinates(only needed if x is a <a href="#">SpatialPolygonsDataFrame</a> ).
units	"cell" or "squaremeter". Indicate if the size units wanted are in number of cells occupied or in square meters(only needed if x is a <a href="#">PresenceAbsence</a> object).

**Value**

The result is a matrix with the range size of each species. If the range size accounts for the earth curvature (Yes or No) or its size unit may differ for each argument combination:

- 1) SpatialPolygonsDataFrame & geographical = Square meters. Yes.
- 2) SpatialPolygonsDataFrame & planar = same units as the coordinates. No.
- 3) PresenceAbsence & cell = number of cells. No.
- 4) PresenceAbsence & squaremeter = Square meters. Yes.

**Author(s)**

Bruno Vilela

**Examples**

```
## Not run:
# SpatialPolygonsDataFrame & geographical
data(Phyllomedusa)
rangesize <- lets.rangesize(x = Phyllomedusa,
                           coordinates = "geographic")

# SpatialPolygonsDataFrame & planar
rangesize2 <- lets.rangesize(x = Phyllomedusa,
                             coordinates = "planar")

# PresenceAbsence & cell
data(PAM)
rangesize3 <- lets.rangesize(x = PAM,
                             units = "cell")

# PresenceAbsence & squaremeter
rangesize4 <- lets.rangesize(x = PAM,
                             units = "squaremeter")

## End(Not run)
```

---

lets.shFilter

*Filter species' shapefiles based on its presence, origin, and season*

---

**Description**

Filter species shapefiles by origin, presence, and seasonal type (following IUCN types: <http://www.iucnredlist.org/technical-documents/spatial-data>, see metadata).

**Usage**

```
lets.shFilter(shapes, presence = NULL, origin = NULL, seasonal = NULL)
```

**Arguments**

shapes	Object of class SpatialPolygonsDataFrame (see function <a href="#">readShapePoly</a> to open these files).
presence	A vector with the code numbers for the presence type to be maintained.
origin	A vector with the code numbers for the origin type to be maintained.
seasonal	A vector with the code numbers for the seasonal type to be maintained.

**Details**

Presence codes: (1) Extant, (2) Probably Extant, (3) Possibly Extant, (4) Possibly Extinct, (5) Extinct (post 1500) & (6) Presence Uncertain.

Origin codes: (1) Native, (2) Reintroduced, (3) Introduced, (4) Vagrant & (5) Origin Uncertain.

Seasonal codes: (1) Resident, (2) Breeding Season, (3) Non-breeding Season, (4) Passage & (5) Seasonal Occurrence Uncertain.

More info in the shapefiles' metadata.

**Value**

The result is the shapefile(s) filtered according to the selected types. If the filters remove all polygons, the result will be NULL.

**Author(s)**

Bruno Vilela

**See Also**

[plot.PresenceAbsence](#)

[lets.presab](#)

[lets.presab.birds](#)

---

lets.subsetPAM

*Subset a PresenceAbsence object based on species names*

---

**Description**

Subset a PresenceAbsence object based on species character vector provided by the user.

**Usage**

```
lets.subsetPAM(x, names, remove.cells = TRUE)
```

**Arguments**

x	A <a href="#">PresenceAbsence</a> object.
names	Character vector with species names to subset the PresenceAbsence object.
remove.cells	Logical, if TRUE the final matrix will not contain cells in the grid with a value of zero (i.e. sites with no species present).

**Value**

The result is an object of class PresenceAbsence subseted.

**Author(s)**

Bruno Vilela

**See Also**

[plot.PresenceAbsence](#)  
[lets.presab.birds](#)

**Examples**

```
## Not run:
data(PAM)
# PAM before subset
plot(PAM, xlab = "Longitude", ylab = "Latitude",
      main = "Phyllomedusa species richness")

# Subset PAM to the first 20 species
PAMsub <- lets.subsetPAM(PAM, PAM[[3]][1:20])
plot(PAMsub, xlab = "Longitude", ylab = "Latitude",
      main = "Phyllomedusa species richness")

## End(Not run)
```

---

lets.summarizer	<i>Summarize variable(s) values in a presence-absence matrix within species' ranges</i>
-----------------	---

---

**Description**

Based on a Presence-Absence matrix with added variables (see [lets.addvar](#)), this function summarizes the values of such variable(s) per species (across the species' occupied cells. i.e. within their ranges).

**Usage**

```
lets.summarizer(x, pos, xy = TRUE, fun = mean)
```

**Arguments**

x	Presence-absence matrix with variables added.
pos	Column position of the variables of interest.
xy	Logical, if TRUE the input matrix contains geographic coordinates in the first two columns.
fun	Function to be used to summarize the variable per species.

**Author(s)**

Bruno Vilela & Fabricio Villalobos

**References**

Villalobos, F. and Arita, H.T. 2010. The diversity field of New World leaf-nosed bats (Phyllostomidae). *Global Ecology and Biogeography*. 19, 200-211.

**See Also**

[lets.addvar](#)  
[lets.addpoly](#)  
[lets.field](#)

**Examples**

```
## Not run:  
data(PAM)  
data(temp)  
pamvar <- lets.addvar(PAM, temp)  
resu <- lets.summarizer(x = pamvar, pos = ncol(pamvar),  
                      xy = TRUE)  
  
## End(Not run)
```

---

lets.transf

*Transform values of a vector*

---

**Description**

Transform each element of a vector.

**Usage**

```
lets.transf(x, y, z, NUMERIC = TRUE)
```



**Arguments**

x	A vector to be transformed.
y	levels to be transformed.
z	The value to be attributed to each level (same order as y).
NUMERIC	logical, if TRUE z will be considered numbers.

**Value**

Return a vector with changed values.

**Author(s)**

Bruno Vilela

**Examples**

```
## Not run:
status <- sample(c("EN", "VU", "NT", "CR", "DD", "LC"), 30, replace=TRUE)
TE <- "Threatened"
NT <- "Non-Threatened"
new <- c(TE, TE, NT, TE, "Data Deficient", NT)
old <- c("EN", "VU", "NT", "CR", "DD", "LC")
statustrans <- lets.transf(status, old, new, NUMERIC=FALSE)

## End(Not run)
```

---

PAM

*PresenceAbsence object for frogs of Phyllomedusa genus*

---

**Description**

PresenceAbsence object obtained using the function `lets.presab` in the Geographic distribution of the Southern American frog genus [Phyllomedusa](#).

**Usage**

```
data(PAM)
```

**Source**

IUCN - <http://www.iucnredlist.org/>. 2014.

---

 Phyllomedusa

*Geographic distribution of Phyllomedusa genus*


---

**Description**

Geographic distribution of the Southern American frog genera Phyllomedusa. Data was modified from IUCN (<http://www.iucnredlist.org/>, downloaded in 05/2014). There are 32 species and 46 polygons.

**Usage**

```
data(Phyllomedusa)
```

**Source**

IUCN - <http://www.iucnredlist.org/>. 2014.

---

 plot.PresenceAbsence *Plot an object of class PresenceAbsence*


---

**Description**

Plots species richness map from an object of class PresenceAbsence or a particular species' map.

**Usage**

```
## S3 method for class 'PresenceAbsence'
plot(x, name = NULL, world = TRUE, col_rich = NULL, ...)
```

**Arguments**

x	An object of class <a href="#">PresenceAbsence</a> .
name	A character specifying a species to be plotted instead of the complete species richness map.
world	If TRUE a map of political divisions (countries) is added to the plot.
col_rich	Color function (e.g. <a href="#">rainbow</a> , <a href="#">heat.colors</a> , <a href="#">colorRampPalette</a> ) to be used in the richness map.
...	Other parameters pass to the plot function.

**Author(s)**

Bruno Vilela

**See Also**

[lets.presab](#)  
[lets.presab.birds](#)

**Examples**

```
## Not run:  
data(PAM)  
plot(PAM)  
plot(PAM, xlab = "Longitude", ylab = "Latitude",  
      main = "Phyllomedusa species richness")  
plot(PAM, name = "Phyllomedusa atelopoides")  
plot(PAM, name = "Phyllomedusa azurea")  
  
## End(Not run)
```

---

PresenceAbsence	<i>PresenceAbsence Class</i>
-----------------	------------------------------

---

**Description**

The PresenceAbsence is a new S3 object class created and used inside the [letsR](#) package. This object class is used to store information on species distribution within a geographic domain in the form of a presence-absence matrix. In addition, the PresenceAbsence object also contains other essential information (e.g. user-defined grid cell system, including resolution, projection, datum, and extent) necessary for other analysis performed with the package's functions.

**Details****Creating a PresenceAbsence object**

A PresenceAbsence object can be generated using the following functions:

- [lets.presab](#)
- [lets.presab.birds](#)
- [lets.presab.points](#)

**The PresenceAbsence information**

The result is a list object of class PresenceAbsence that includes the following objects:

- Presence\_and\_Absence\_Matrix: A matrix of species' presence(1) and absence(0) information. The first two columns contain the longitude (x) and latitude (y) of the cells' centroid (from the gridded domain used);
- Richness\_Raster: A raster containing species richness information across the geographic domain, which can be used to map the observed geographic gradient in species richness;
- Species\_name: A character vector with species' names contained in the matrix.

Each of the objects can be obtained using the standard subsetting operators that are commonly

applied to a list object (i.e. '[' and '\$').

### letsR functions applied to a PresenceAbsence object

The following functions from the letsR package can be directly applied to a PresenceAbsence:

- `lets.addpoly`
- `lets.addvar`
- `lets.dismat`
- `lets.field`
- `lets.gridirizer`
- `lets.iucn`
- `lets.iucn.ha`
- `lets.iucn.his`
- `lets.maplizer`
- `lets.midpoint`
- `lets.overlap`
- `lets.pamcrop`
- `lets.rangesize`

### Generic functions applied to a PresenceAbsence object

The following generic functions can be directly applied to the PresenceAbsence object.

- `print` (`print.PresenceAbsence`)
- `summary` (`summary.PresenceAbsence`)
- `plot` (`plot.PresenceAbsence`)

---

`print.PresenceAbsence` *Print for object of class PresenceAbsence*

---

### Description

Print for objects of class PresenceAbsence.

### Usage

```
## S3 method for class 'PresenceAbsence'  
print(x, ...)
```

### Arguments

`x` an object of class `PresenceAbsence`.  
`...` Other print parameters.

### Author(s)

Bruno Vilela

---

print.summary.PresenceAbsence

*Print summary for object of class PresenceAbsence*

---

### Description

Print summary for objects of class PresenceAbsence.

### Usage

```
## S3 method for class 'PresenceAbsence'  
print.summary(x, ...)
```

### Arguments

x                    an object of class [PresenceAbsence](#).  
...                   Other print parameters.

### Author(s)

Bruno Vilela

---

summary.PresenceAbsence

*Summary for object of class PresenceAbsence*

---

### Description

Summary for objects of class PresenceAbsence.

### Usage

```
## S3 method for class 'PresenceAbsence'  
summary(object, ...)
```

### Arguments

object               an object of class [PresenceAbsence](#).  
...                   Other summary parameters.

### Author(s)

Bruno Vilela

---

temp

*Average temperature raster for the world.*

---

**Description**

Average temperature raster in Celsius degrees(multiplied by 100) for the world in 10 arc min of resolution. Data was modified from WorldClim (<http://worldclim.com/>, downloaded in 05/2014).

**Usage**

```
data(temp)
```

**Source**

Hijmans, R.J., S.E. Cameron, J.L. Parra, P.G. Jones and A. Jarvis, 2005. Very high resolution interpolated climate surfaces for global land areas. *International Journal of Climatology* 25: 1965-1978.

# Index

## \*Topic **package**

letsR-package, 2

colorRampPalette, 34

heat.colors, 34

hist, 6

IUCN, 3

lets.addpoly, 4, 5, 32, 36

lets.addvar, 3, 4, 5, 6, 31, 32, 36

lets.classvar, 6

lets.correl, 7

lets.distmat, 8, 36

lets.field, 9, 32, 36

lets.gridirizer, 10, 36

lets.iucn, 3, 11, 13–15, 36

lets.iucn.ha, 3, 12, 12, 14, 36

lets.iucn.his, 3, 12, 13, 13, 36

lets.iucncont, 15

lets.maplizer, 16, 36

lets.midpoint, 17, 36

lets.overlap, 18, 36

lets.pamcrop, 19, 36

lets.presab, 3–5, 10, 17–19, 20, 23, 24, 27, 30, 33, 35

lets.presab.birds, 3–5, 10, 11, 17–19, 22, 22, 25, 27, 30, 31, 35

lets.presab.grid, 24

lets.presab.points, 3, 26, 35

lets.rangesize, 28, 36

lets.shFilter, 22, 24, 25, 27, 29

lets.subsetPAM, 30

lets.summarizer, 31

lets.transf, 32

letsR, 35

letsR (letsR-package), 2

letsR-package, 2

Phyllomedusa, 33, 34

plot.PresenceAbsence, 11, 19, 22, 24, 25, 27, 30, 31, 34, 36

PresenceAbsence, 3–5, 8–13, 16–19, 21, 24, 27, 28, 31, 34, 35, 36, 37

PresenceAbsence-class  
(PresenceAbsence), 35

print.PresenceAbsence, 36, 36

print.summary.PresenceAbsence, 37

rainbow, 34

readShapePoly, 19, 20, 25, 30

summary.PresenceAbsence, 36, 37

temp, 38

PAM, 33