{rrapply}: Revisiting R-base rapply()

The minimal rrapply-package contains a single function rrapply(), providing an extended implementation of R-base’s rapply() function. rrapply() recursively applies a function f to elements of a nested list and controls how to structure the returned result.

Function signature

```
rrapply(
  object,  # a "list-like" object;
  condition,  # a condition function for application of f;
  f,  # a function to recursively apply to each list element;
  classes = c("list", "ANY"),  # classes to which f is applied, can include "list" or "data.frame";
  deflt = NULL,  # a default return value;
  how = "melt",  # how to structure the result;
  ...  # additional arguments for f and condition functions;
  options)  # additional options for how;
```

Example data

```
library(rrapply)
# data: renewable energy per country 2016
# as % of total energy consumption
data("renewable_energy_by_country")
data("pokedex")
```

How to structure the result

- **how = "replace"**
  
  replaces elements x satisfying condition and classes by f(x) and others by deflt maintaining list structure:
  
  ```
  # replace all missing values by 0
  rrapply(renewable_energy_by_country, condition = \( \text{!is.na}(x) \), deflt = 0, how = "replace")
  ```
  
- **how = "unlist"**
  
  similar to how = "list" unlisting the returned result:
  
  ```
  # replace missing values by 0 and unlist
  rrapply(renewable_energy_by_country, classes = "numeric", deflt = 0, how = "unlist")
  ```
  
- **how = "prune"**
  
  similar to how = "prune" returning a pruned list.
  
  ```
  # prune all missing values and maintain list structure
  rrapply(renewable_energy_by_country, condition = \( \text{!is.na}(x) \), how = "prune")
  ```
  
- **how = "flatten"**
  
  similar to how = "prune" returning a flattened list.
  
  ```
  # flatten list
  rrapply(renewable_energy_by_country, condition = \( \text{!is.na}(x) \), how = "flatten")
  ```
  
- **how = "melt"**
  
  similar to how = "prune" returning a melted list.
  
  ```
  # prune all missing values and melt list
  rrapply(renewable_energy_by_country, condition = \( \text{!is.na}(x) \), how = "melt")
  ```
  
- **how = "names"**
  
  similar to how = "recurse" replacing the name of element x by f(x) instead of its content using classes = c("List", "ANY") by default:
  
  ```
  # recursively capitalize all names in list
  rrapply(renewable_energy_by_country, f = \( \text{toupper}(\text{xname}) \), how = "names")
  ```
  
Special arguments .xname, .xpos and .xparents

The f and condition functions accept four special arguments in addition to the principal argument:

- **.xname** evaluates to the name of the current list element:
  
  ```
  # filter list elements by name
  rrapply(renewable_energy_by_country, condition = \( \text{xname} == \text{"Belgium"} \), how = "prune")
  ```
  
- **.xpos** evaluates to the position of the element in the nested list as an integer vector:
  
  ```
  # return position of element in list
  rrapply(renewable_energy_by_country, condition = \( \text{xname} == \text{"Belgium"} \), f = \( \text{\text{xpos}} \), how = "flatten")
  ```
  
- **.xparents** evaluates to the vector of parent names of the current element.

  ```
  # filter list elements by parent names
  rrapply(renewable_energy_by_country, condition = \( \text{\text{xparents}} \), how = "melt")
  ```

R-package webpage: https://jorischau.github.io/rrapply/