

# Package ‘forestry’

July 22, 2025

**Type** Package

**Title** Reshape Data Tree

**Version** 0.1.1

**Maintainer** Jiena McLellan <jienagu90@gmail.com>

**Description** A series of utility functions to help with  
reshaping hierarchy of data tree, and reform the structure of data tree.

**BugReports** <https://github.com/jienagu/forestry/issues>

**Suggests** knitr, rmarkdown

**VignetteBuilder** knitr

**License** MIT + file LICENSE

**Imports** data.tree

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.0.2

**NeedsCompilation** no

**Author** Jiena McLellan [aut, cre] (ORCID:  
<<https://orcid.org/0000-0002-5578-088X>>),  
Michael Condouris [ctb] (ORCID:  
<<https://orcid.org/0000-0002-8862-4250>>),  
Sai Im [ctb] (ORCID: <<https://orcid.org/0000-0003-1990-5179>>)

**Depends** R (>= 3.5.0)

**Repository** CRAN

**Date/Publication** 2023-11-20 19:20:02 UTC

## Contents

add_child . . . . .	2
assign_attr . . . . .	3
children_sort . . . . .	3
create_nodes . . . . .	4

create_tree . . . . .	5
cumsum_across_level . . . . .	5
cumsum_by_level . . . . .	6
exercise_df . . . . .	7
fill_NA_level . . . . .	7
fixnames . . . . .	8
fix_items . . . . .	8
pre_get_array . . . . .	9
test_df . . . . .	10

## Index 11

---

add_child	<i>Add children node</i>
-----------	--------------------------

---

### Description

Add children node

### Usage

```
add_child(main_tree, x, assign_node)
```

### Arguments

main_tree	the parent tree to be appended with children node
x	xth child
assign_node	appended node as child

### Value

reshaped tree with children assigned

### Examples

```
data("test_df")
data("exercise_df")
test_node <- data.tree::as.Node(test_df)
test_exercise <- data.tree::as.Node(exercise_df)
add_child(main_tree = test_node, x = 4, assign_node = test_exercise )
print(test_node)
```

---

assign_attr	<i>assign attributes to node; work with fix_items function</i>
-------------	--

---

**Description**

assign attributes to node; work with fix\_items function

**Usage**

```
assign_attr(node_from, node_to)
```

**Arguments**

node_from	assigned attributes from
node_to	assigned attributes to

**Value**

a node assigned attributes

**Examples**

```
cell_node1 <- data.tree::Node$new("cell1")
cell_node1$AddChild("A")
cell_node2 <- data.tree::Node$new("cell2")
cell_node2$AddChild("A")
cell_node2$Set(group = c(NA, "A1"))
print(assign_attr(node_from = cell_node1$A, node_to = cell_node2$A), "group")
```

---

children_sort	<i>Sort children nodes with certain order</i>
---------------	---

---

**Description**

Sort children nodes with certain order

**Usage**

```
children_sort(input_node, input_order, mismatch_last = T)
```

**Arguments**

input_node	input node
input_order	children node order
mismatch_last	TRUE: mismatched children nodes are at the bottom; FALSE: mismatched nodes are at the top

**Value**

tree with children nodes sorted with certian order

**Examples**

```
data(test_df)
test_node <- data.tree::as.Node(test_df)
sorted_node <- children_sort(
  input_node = test_node,
  input_order = c("groupB", "groupA"),
  mismatch_last = TRUE)
print(sorted_node)
```

---

create\_nodes

*create a tree with assigned name, children and fields*

---

**Description**

create a tree with assigned name, children and fields

**Usage**

```
create_nodes(tree_name, add_children_count, ...)
```

**Arguments**

tree_name	assign name of tree
add_children_count	assign number of children to this tree
...	parameters that will be passed as fields of this tree

**Value**

a tree with assigned name, children and fields

**Examples**

```
create_nodes(tree_name = "tree1", add_children_count = 3, class = c("A", "B", "C"))
```

---

create\_tree                    *create tree appended with each element of input list as a child*

---

**Description**

create tree appended with each element of input list as a child

**Usage**

```
create_tree(input_list, node_name)
```

**Arguments**

input_list	input list to be made for a tree
node_name	name of the tree

**Value**

a tree with each item of the list as each child

**Examples**

```
data("test_df")
test_node <- data.tree::as.Node(test_df)
new_shape <- create_tree(test_node$children, "new_tree")
print(new_shape, "hc")
```

---

cumsum\_across\_level            *cumulative calculation*

---

**Description**

cumulative calculation

**Usage**

```
cumsum_across_level(input_node, attri_name, level_num)
```

**Arguments**

input_node	tree
attri_name	name of this cummulative count field
level_num	calculate cummulative value cross the level

**Value**

tree with cummulative count

**Examples**

```
data(exercise_df)
exercise_node <- data.tree::as.Node(exercise_df)
test <- cumsum_across_level(input_node = exercise_node,
                           attri_name = "exercise_time",
                           level_num = 3)
print(test, "cumsum_number", "exercise_time", "level")
```

---

cumsum_by_level	<i>calculate cumsum for input level</i>
-----------------	---

---

**Description**

calculate cumsum for input level

**Usage**

```
cumsum_by_level(input_tree, level_num, attri_name)
```

**Arguments**

input_tree	input tree
level_num	level of tree for cumsum
attri_name	name of this cummulative count field

**Value**

tree with calculated cumsum for input level

**Examples**

```
data(exercise_df)
exercise_node <- data.tree::as.Node(exercise_df)
cumsum_by_level(exercise_node, 3, "exercise_time")
```

---

exercise_df	<i>Anonymized sample exercise data</i>
-------------	--

---

**Description**

Anonymized sample exercise data

**Usage**

```
data(exercise_df)
```

**Format**

a data frame ready to convert to a tree

**Author(s)**

Jiena Gu McLellan, 2020-05-26

**Examples**

```
data(exercise_df)
```

---

fill_NA_level	<i>fill missing value of a field across a level with 0</i>
---------------	--

---

**Description**

fill missing value of a field across a level with 0

**Usage**

```
fill_NA_level(input_node, field_name, by_level, fill_with = 0)
```

**Arguments**

input_node	input node
field_name	field for this operation
by_level	across this level
fill_with	fill missing value with this value

**Value**

node with NA filled for the input field at input level

**Examples**

```

data(exercise_df)
exercise_node <- data.tree::as.Node(exercise_df)
result <- fill_NA_level(input_node = exercise_node,
                       field_name = "exercise_time",
                       by_level = 2,
                       fill_with = "quarterly")
print(result, "exercise_time")

```

---

fixnames	<i>numericalize children numeric name to convert JSON object to JSON array</i>
----------	--

---

**Description**

numericalize children numeric name to convert JSON object to JSON array

**Usage**

```
fixnames(x)
```

**Arguments**

x                   input

**Value**

unname numeric names list

**Examples**

```
fixnames(list("1" = 1, "2" = 2))
```

---

fix_items	<i>assign certain children nodes and fill NA for empty fields</i>
-----------	---

---

**Description**

assign certain children nodes and fill NA for empty fields

**Usage**

```
fix_items(fix_vector, input_node)
```



**Arguments**

fix\_vector      children node names to be assigned  
input\_node      the node to be expanded with children's names

**Value**

a node expanded with certain children nodes

**Examples**

```
cell_node2 <- data.tree::Node$new("cell2")
cell_node2$AddChild("B")
cell_node2$AddChild("C")
cell_node2$Set(class = c(NA, "B1", "C1"))
print(cell_node2, "class")
cell_fixed_items <- fix_items(fix_vector = c("A", "B", "C", "D"), input_node = cell_node2)
print(cell_fixed_items, "class")
```

---

pre_get_array	<i>numericalize children numeric name to convert JSON object to JSON array</i>
---------------	--

---

**Description**

numericalize children numeric name to convert JSON object to JSON array

**Usage**

```
pre_get_array(x)
```

**Arguments**

x                    input list

**Value**

unname numeric names list which is prepared to convert to JSON array

**Examples**

```
demo_list <- list("1" = 1, "2" = 2, list("1" = 1, "2" = 2))
pre_get_array(demo_list)
```

---

test\_df

*Anonymized sample data*

---

**Description**

Anonymized sample data

**Usage**

```
data(test_df)
```

**Format**

a data frame ready to convert to a tree

**Author(s)**

Jiena Gu McLellan, 2020-05-26

**Examples**

```
data(test_df)
```

# Index

## \* datasets

exercise\_df, 7  
test\_df, 10

add\_child, 2  
assign\_attr, 3

children\_sort, 3  
create\_nodes, 4  
create\_tree, 5  
cumsum\_across\_level, 5  
cumsum\_by\_level, 6

exercise\_df, 7

fill\_NA\_level, 7  
fix\_items, 8  
fixnames, 8

pre\_get\_array, 9

test\_df, 10