

# Package ‘petrinetR’

March 8, 2019

**Type** Package

**Title** Building, Visualizing, Exporting and Replaying Petri Nets

**Version** 0.2.1

**Date** 2019-03-08

**Description** Functions for the construction of Petri Nets. Petri Nets can be replayed by firing enabled transitions.  
Silent transitions will be hidden by the execution handler. Also includes functionalities for the visualization of Petri Nets and export of Petri Nets to PNML (Petri Net Markup Language) files.

**License** GPL-3

**Depends** R(>= 3.0.0)

**LazyData** true

**Imports** dplyr, visNetwork, DiagrammeR, xml2, purrr

**RoxygenNote** 6.1.1

**URL** <https://www.bupar.net>

**BugReports** <https://github.com/gertjanssenswillen/petrinetR/issues>

**NeedsCompilation** no

**Author** Gert Janssenswillen [aut, cre]

**Maintainer** Gert Janssenswillen <gert.janssenswillen@uhasselt.be>

**Repository** CRAN

**Date/Publication** 2019-03-08 11:30:03 UTC

## R topics documented:

create_PN . . . . .	2
enabled . . . . .	3
enabled_transition . . . . .	3
execute . . . . .	4
flows . . . . .	4
is_place . . . . .	5

is_transition . . . . .	5
marking . . . . .	5
n_places . . . . .	6
parse_trace . . . . .	7
parse_trace . . . . .	7
part_of . . . . .	8
petrinetR . . . . .	8
places . . . . .	8
post_set . . . . .	9
pre_set . . . . .	9
print.petrinet . . . . .	10
read_PN . . . . .	10
render_PN . . . . .	10
transitions . . . . .	11
tree_to_PN . . . . .	11
visNetwork_from_PN . . . . .	11

<b>Index</b>	<b>12</b>
--------------	-----------

---

create_PN	<i>Create Petri Net</i>
-----------	-------------------------

---

## Description

Function to create a petri net by specifying a set of places, transitions, flows and a marking.

## Usage

```
create_PN(places, transitions, flows, marking)
```

## Arguments

places	A vector of unique places.
transitions	A vector of unique transitions.
flows	A data.frame of flows, with columns named "from" and "to".
marking	The names of the places to be marked.

## Examples

```
create_PN("place_1",
"transition_1",
data.frame(from = "place_1",to = "transition_1"),
marking = "place_1")
```

---

enabled	<i>Enabled transitions</i>
---------	----------------------------

---

**Description**

List the enabled transitions in a marked Petri Net. Silent transitions, i.e. starting with "inv\_" or "tau" are assumed to be able to fire silently, thereby possible enabling other transitions.

**Usage**

enabled(PN)

**Arguments**

PN            A Petri Net

---

enabled_transition	<i>Enabled Transition</i>
--------------------	---------------------------

---

**Description**

Check if a transition is currently enabled

**Usage**

enabled\_transition(PN, transition)

**Arguments**

PN            A Petri Net  
transition    A Transition

---

execute	<i>Execute</i>
---------	----------------

---

**Description**

Executes (fire) an enabled transition and returns the Petri Net with the New marking. If the transition is enabled via the firing of silent transition (i.e. starting with "inv\_" of "tau"), it will fire these first. If the transition is not enabled, it will return FALSE.

**Usage**

```
execute(PN, transition)
```

**Arguments**

PN	A Petri Net
transition	The transition to be fired

---

flows	<i>Flows</i>
-------	--------------

---

**Description**

Extracts the flows from a Petri Net

**Usage**

```
flows(PN)
```

**Arguments**

PN	A Petri Net
----	-------------

---

is_place	<i>Is place</i>
----------	-----------------

---

**Description**

Check if a place is part of a petri net

**Usage**

```
is_place(place, PN)
```

**Arguments**

place	A place
PN	A Petri Net

---

---

is_transition	<i>Is transition</i>
---------------	----------------------

---

**Description**

Check if a transition is part of a petri net

**Usage**

```
is_transition(transition, PN)
```

**Arguments**

transition	A transition
PN	A Petri Net

---

---

marking	<i>Marking</i>
---------	----------------

---

**Description**

Get the current marking of a Petri Net

**Usage**

```
marking(PN)
```

**Arguments**

PN	A Petri Net
----	-------------

---

n\_places

*Utils*

---

## Description

Several auxilliary functions for Petri Net objects.

## Usage

n\_places(PN)

n\_transitions(PN)

n\_flows(PN)

n\_nodes(PN)

nodes(PN)

rename\_transitions(PN, .f, ...)

rename\_places(PN, .f, ...)

add\_places(PN, .p)

add\_transitions(PN, .t)

add\_flows(PN, .flows)

## Arguments

PN	A petri net
.f	A function name to apply for renaming
...	Additional arguments
.p	A character vector of places
.t	A character vector of transitions
.flows	A data.frame with a to and from column

---

parse_trace	<i>Parse (logical)</i>
-------------	------------------------

---

**Description**

Tests whether a sequence of transitions can be fired by a Petri Net. If so returns TRUE, otherwise FALSE.

**Usage**

parse\_trace(PN, trace)

**Arguments**

PN	A Petri Net
trace	A sequence of transitions, stored in a vector.

---

parse_trace	<i>Parse</i>
-------------	--------------

---

**Description**

Parses a sequence of transitions. If possible returns the Petri Net with the updated marking. Otherwise returns FALSE

**Usage**

parse\_trace(PN, trace)

**Arguments**

PN	A Petri Net
trace	A sequence of transitions, stored in a vector.

part\_of                      *Part of*

---

**Description**

Check if a node is part of a petri net

**Usage**

part\_of(node, PN)

**Arguments**

node	A node
PN	A Petri Net

---

petrinetR                      *petrinetR - Building, visualizing, exporting and replaying Petri Nets*

---

**Description**

Functions for the construction of Petri Nets. Petri Nets can be replayed by firing enabled transitions. Silent transitions will be hidden by the execution handler. Also includes functionalities for the visualization of Petri Nets and export of Petri Nets to PNML-files.

---

places                          *Places*

---

**Description**

Extracts the places from a Petri Net

**Usage**

places(PN)

**Arguments**

PN	A Petri Net
----	-------------



---

post_set	<i>Postset</i>
----------	----------------

---

**Description**

Get the postset of a transition or place in a Petri Net

**Usage**

post\_set(PN, node)

**Arguments**

PN	A Petri Net
node	A place or transition in the petri net

---

pre_set	<i>Preset</i>
---------	---------------

---

**Description**

Get the preset of a transition or place in a Petri Net

**Usage**

pre\_set(PN, node)

**Arguments**

PN	A Petri Net
node	A place or transition in the petri net

---

print.petrinet	<i>Generic print function for petrinet</i>
----------------	--

---

**Description**

Generic print function for petrinet

**Usage**

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

**Arguments**

x	petrinet object
...	Additional Arguments

---

read_PN	<i>Read PNML</i>
---------	------------------

---

**Description**

Function

**Usage**

```
read_PN(file)
```

**Arguments**

file	Path to .pnml file
------	--------------------

---

render_PN	<i>Render Petri Net</i>
-----------	-------------------------

---

**Description**

Function

**Usage**

```
render_PN(PN)
```

**Arguments**

PN	A petri net
----	-------------

---

transitions	<i>Transitions</i>
-------------	--------------------

---

**Description**

Extracts the transitions from a Petri Net

**Usage**

transitions(PN)

**Arguments**

PN	A Petri Net
----	-------------

---

tree_to_PN	<i>tree_to_PN</i>
------------	-------------------

---

**Description**

Create of petri net from a process tree.

**Usage**

tree\_to\_PN(tree)

**Arguments**

tree	The process tree to be converted
------	----------------------------------

---

visNetwork_from_PN	<i>VisNetwork from PN</i>
--------------------	---------------------------

---

**Description**

Visualize a Petri Net with an interactive network

**Usage**

visNetwork\_from\_PN(PN)

**Arguments**

PN	Petri Net to visualize
----	------------------------

# Index

`add_flows (n_places)`, 6  
`add_places (n_places)`, 6  
`add_transitions (n_places)`, 6  
  
`create_PN`, 2  
  
`enabled`, 3  
`enabled_transition`, 3  
`execute`, 4  
  
`flows`, 4  
  
`is_place`, 5  
`is_transition`, 5  
  
`marking`, 5  
  
`n_flows (n_places)`, 6  
`n_nodes (n_places)`, 6  
`n_places`, 6  
`n_transitions (n_places)`, 6  
`nodes (n_places)`, 6  
  
`parse_trace`, 7  
`parsel_trace`, 7  
`part_of`, 8  
`petrinetR`, 8  
`petrinetR-package (petrinetR)`, 8  
`places`, 8  
`post_set`, 9  
`pre_set`, 9  
`print.petrinet`, 10  
  
`read_PN`, 10  
`rename_places (n_places)`, 6  
`rename_transitions (n_places)`, 6  
`render_PN`, 10  
  
`transitions`, 11  
`tree_to_PN`, 11  
  
`visNetwork_from_PN`, 11