# Package 'nomnoml'

June 13, 2019

**Type** Package

**Title** Sassy 'UML' Diagrams

**Version** 0.1.0

**Maintainer** Javier Luraschi <javier@rstudio.com>

**Description** A tool for drawing sassy 'UML' diagrams based on a simple syntax, see <http://www.nomnoml.com>. Supports styling, R Markdown and exporting diagrams in the PNG format.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**Depends** R (>= 3.1.2)

**Imports** htmlwidgets, png, webshot

**RoxygenNote** 6.1.1

**Suggests** knitr, rmarkdown

**NeedsCompilation** no

**Author** Javier Luraschi [aut, cre], Daniel Kallin [ctb] (nomnoml.js library, http://nomnoml.com)

**Repository** CRAN

**Date/Publication** 2019-06-13 16:20:07 UTC

## R topics documented:

nomnoml *Render Diagram*

## Description

Renders a 'nomnoml' diagram as an 'htmlwidget' or saves it as a 'png' image.

## Usage

```
nomnoml(code = "[Hello]-[World!]", png = NULL, width = NULL,
  height = NULL, ...)
```

## Arguments

| | |
|---|---|
| code | The nomnoml diagram code. |
| png | Optional file name to export diagram as 'png'. |
| width | Optional width in pixels for the exported 'png'. |
| height | Optional height in pixels for the exported 'png'. |
| ... | Additional parameters. |

## Details

The 'nomnoml' syntax is simple and intuitive, a "Hello World" example can be rendered as an 'htmlwidget' as follows:

```
nomnoml::nomnoml("[Hello]-[World!]", "orange")
```

You can also render as a 'png' file with specific dimensions:

```
nomnoml::nomnoml("[Hello]-[World!]", "hello.png", 600, 100)
```

Still, complex diagrams can be defined by combining multiple association types, classifier types, directives and custom classifier styles.

You can also use of the `nomnoml` 'knitr' chunk to render inline diagrams in R Markdown documents.

## Association Types

```
association -
association ->
association <->
dependency -->
dependency <-->
generalization -:>
generalization <:-
implementation --:>
implementation <:--
```

```
composition +-
composition +->
aggregation o-
aggregation o->
note --
hidden -/-
```

### Classifier Types

```
[name]
[<abstract> name]
[<instance> name]
[<note> name]
[<reference> name]
[<package> name]
[<frame> name]
[<database> name]
[<start> name]
[<end> name]
[<state> name]
[<choice> name]
[<input> name]
[<sender> name]
[<receiver> name]
[<transceiver> name]
[<actor> name]
[<usecase> name]
[<label> name]
[<hidden> name]
```

### Directives

```
#arrowSize: 1
#bendSize: 0.3
#direction: down | right
#gutter: 5
#edgeMargin: 0
#edges: hard | rounded
#fill: #eee8d5; #fdf6e3
```

```
#fillArrows: false
#font: Calibri
#fontSize: 12
#leading: 1.25
#lineWidth: 3
#padding: 8
#spacing: 40
#stroke: #33322E
#title: filename
#zoom: 1
```

### Custom Classifier Styles

A directive that starts with . define a classifier style. The style is written as a space separated list of modifiers and key/value pairs.

```
#.box: fill=#8f8 dashed
#.blob: visual=ellipse
[<box> GreenBox]
[<blob> HideousBlob]
```

Available key/value pairs are:

```
fill=(any css color)
stroke=(any css color)
align=center
align=left
direction=right
direction=down
visual=actor
visual=class
visual=database
visual=ellipse
visual=end
visual=frame
visual=hidden
visual=input
visual=none
visual=note
visual=package
visual=receiver
```

visual=rhomb

visual=roundrect

visual=sender

visual=start

visual=transceiver

Available modifiers are:

bold

underline

italic

dashed

empty

## Examples

```
# Render simple diagram:
nomnoml::nomnoml("[Hello]-[World!]")

# Render complex diagram:
nomnoml::nomnoml("
  #stroke: #a86128
  [<frame>Decorator pattern|
    [<abstract>Component||+ operation()]
    [Client] depends --> [Component]
    [Decorator|- next: Component]
    [Decorator] decorates -- [ConcreteComponent]
    [Component] <:- [Decorator]
    [Component] <:- [ConcreteComponent]
  ]")
```

---

`nomnoml-shiny` *Shiny bindings for nomnoml*

---

## Description

Output and render functions for using nomnoml within Shiny applications and interactive Rmd documents.

## Usage

```
nomnomlOutput(outputId, width = "100%", height = "400px")

renderNomnoml(expr, env = parent.frame(), quoted = FALSE)
```

## Arguments

| | |
|---|---|
| `outputId` | output variable to read from |
| `width, height` | Must be a valid CSS unit (like `'100%'`, `'400px'`, `'auto'`) or a number, which will be coerced to a string and have `'px'` appended. |
| `expr` | An expression that generates a nomnoml |
| `env` | The environment in which to evaluate `expr`. |
| `quoted` | Is `expr` a quoted expression (with `quote()`)? This is useful if you want to save an expression in a variable. |

# Index