Package: Rmaze (via r-universe)

May 20, 2026

Type Package

Title A package for generating mazes

Version 0.1.0

Date 2026-08-08

Maintainer Vesna Memisevic <vesna@vesnam.com>

Description A package for generating mazes in R.

License MIT + file LICENSE

LazyData TRUE

BugReports https://github.com/Vessy/Rmaze/issues

URL https://github.com/Vessy/Rmaze

Imports stringr (>= 1.0.0), igraph (>= 1.0.1), ggplot2 (>= 2.1.0), shiny (>= 0.13.2),

RoxygenNote 5.0.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

Config/pak/sysreqs cmake libglpk-dev make libicu-dev libuv1-dev libxml2-dev zlib1g-dev

Repository https://kbroman.r-universe.dev

Date/Publication 2026-05-09 20:51:26 UTC

RemoteUrl https://github.com/kbroman/Rmaze

RemoteRef HEAD

RemoteSha 0dfa0848350ee89d9a9287cb740641e6c60d5500

Contents

`makeGraph` *Create a connected maze graph that represents a rectangular grid. Nodes represent cells and edges betweem nodes represent wall sites. All walls are initially on.*

Description

Create a connected maze graph that represents a rectangular grid. Nodes represent cells and edges betweem nodes represent wall sites. All walls are initially on.

Usage

```
makeGraph(nrows = 0, ncols = 0)
```

Arguments

`nrows`	maze hight (number of rows); default value set to 0.
`ncols`	maze width (number of columns); default value set to 0.

Value

This function creates and returns a maze graph that represents a rectangular grid that matches user specified maze dimensions. Nodes in the graph will be named in the format Aij, where i corresponds to a row number and j corresponds to a column number.

Examples

```
maze1 <- makeGraph(10, 10)
maze1 <- makeMaze_dfs(maze1)
plotMaze(maze1, 10, 10)
```

makeImperfect *Given a perfect maze graph, create an imperfect maze graph.*

Description

Given a perfect maze graph, create an imperfect maze graph.

Usage

```
makeImperfect(gD = NA, ptc = 20, inShiny = FALSE)
```

Arguments

gD	an existing maze graph object.
ptc	percentage of walls to randomly add or remove; default value is 10(percent).
inShiny	a flag that marks whether the function is called from a shiny app or console.

Value

A maze graph object containing an imperfect maze.

Examples

```
maze1 <- makeGraph(10, 10)
maze1 <- makeMaze_dfs(maze1)
plotMaze(maze1, 10, 10)
maze1 <- makeImperfect(maze1)
plotMaze(maze1, 10, 10)
```

makeMaze_dfs *Create a maze uzing a randomized version of depth-first search algorithm (recursive backtracker).*

Description

Create a maze uzing a randomized version of depth-first search algorithm (recursive backtracker).

Usage

```
makeMaze_dfs(gD = NA, stepBystep = FALSE, nrows = 0, ncols = 0,
  inShiny = FALSE)
```

Arguments

gD	an existing maze graph object.
stepBystep	a flag that will allow a step by step plot of maze creation
nrows	maze hight (number of rows); required only for the step by step plot; default value set to 0.
ncols	maze width (number of columns); required only for the step by step plot; default value set to 0.
inShiny	a flag that marks whether the function is called from a shiny app or console

Value

Given a connected maze graph (with all walls on), this function creates a maze (removes some walls) using depth-first search algorithm (recursive backtracker), and returns the resulting maze graph.

Examples

```
maze1 <- makeGraph(10, 10)
maze1 <- makeMaze_dfs(maze1)
plotMaze(maze1, 10, 10)
```

---

makeMaze_kruskal *Create a maze uzing a randomized Kruskal's algorithm.*

Description

Create a maze uzing a randomized Kruskal's algorithm.

Usage

```
makeMaze_kruskal(gD = NA, stepBystep = FALSE, nrows = 0, ncols = 0,
  inShiny = FALSE)
```

Arguments

gD	an existing maze graph object.
stepBystep	a flag that will allow a step by step plot of maze creation
nrows	maze hight (number of rows); required only for the step by step plot; default value set to 0.
ncols	maze width (number of columns); required only for the step by step plot; default value set to 0.
inShiny	a flag that marks whether the function is called from a shiny app or console

Value

Given a connected maze graph (with all walls on), this function creates a maze (removes some walls) using depth-first search algorithm (recursive backtracker), and returns the resulting maze graph.

Examples

```
maze1 <- makeGraph(10, 10)
maze1 <- makeMaze_kruskal(maze1)
plotMaze(maze1, 10, 10)
```

---

makeMaze_prim *Create a maze uzing a randomized Prim's algorithm.*

Description

Create a maze uzing a randomized Prim's algorithm.

Usage

```
makeMaze_prim(gD = NA, stepBystep = FALSE, nrows = 0, ncols = 0,
  inShiny = FALSE)
```

Arguments

gD	an existing maze graph object.
stepBystep	a flag that will allow a step by step plot of maze creation
nrows	maze hight (number of rows); required only for the step by step plot; default value set to 0.
ncols	maze width (number of columns); required only for the step by step plot; default value set to 0.
inShiny	a flag that marks whether the function is called from a shiny app or console

Value

Given a connected maze graph (with all walls on), this function creates a maze (removes some walls) using depth-first search algorithm (recursive backtracker), and returns the resulting maze graph.

Examples

```
maze1 <- makeGraph(10, 10)
maze1 <- makeMaze_prim(maze1)
plotMaze(maze1, 10, 10)
```

plotMaze *Given a maze graph, plot a maze.*

Description

Given a maze graph, plot a maze.

Usage

```
plotMaze(gD = NA, nrows = 0, ncols = 0, inShiny = FALSE)
```

Arguments

gD	an existing maze graph object.
nrows	maze hight (number of rows); default value set to 0.
ncols	maze width (number of columns); default value set to 0.
inShiny	a flag that marks whether the function is called from a shiny app or console

Value

This function uses ggplot to plot a maze. Currently, maze enterance and exit points (cells/nodes) are fixed.

Examples

```
maze1 <- makeGraph(10, 10)
maze1 <- makeMaze_dfs(maze1)
plotMaze(maze1, 10, 10)
```

plotMazeSolution *Plot maze solution (path)*

Description

Plot maze solution (path)

Usage

```
plotMazeSolution(gD = NA, nrows = 0, ncols = 0, inShiny = FALSE)
```

Arguments

gD	an existing maze graph object.
nrows	maze hight (number of rows); default value set to 0.
ncols	maze width (number of columns); default value set to 0.
inShiny	a flag that marks whether the function is called from a shiny app or console.

Value

The maze solution is found as the shortest path between the start and end maze cells (currently, start and end points of the maze are fixed). The function uses ggplot to plot a maze.

Examples

```
maze1 <- makeGraph(10, 10)
maze1 <- makeMaze_dfs(maze1)
plotMaze(maze1, 10, 10)
plotMazeSolution(maze1, 10, 10)
```

---

runExample *A function to start a Shiny app that comes with the package. Code from: http://www.r-bloggers.com/supplementing-your-r-package-with-a-shiny-app-2/*

Description

A function to start a Shiny app that comes with the package. Code from: http://www.r-bloggers.com/supplementing-your-r-package-with-a-shiny-app-2/

Usage

```
runExample()
```

Examples

```
Rmaze::runExample()
```

---

stepByStepMAze *Plot maze solution (path)*

Description

Plot maze solution (path)

Usage

```
stepByStepMAze(gD = NA, nrows = 0, ncols = 0, inShiny = FALSE)
```

Arguments

gD	an existing maze graph object.
nrows	maze hight (number of rows); default value set to 0.
ncols	maze width (number of columns); default value set to 0.
inShiny	a flag that marks whether the function is called from a shiny app or console.

Value

The maze solution is found as the shortest path between the start and end maze cells (currently, start and end points of the maze are fixed). The function uses ggplot to plot a maze.

Examples

```
maze1 <- makeGraph(10, 10)
maze1 <- makeMaze_dfs(maze1, stepBystep = TRUE, nrows=10, ncols=10)
```

Index