Package ‘FourScores’

October 12, 2022

Title A Game for Human vs. Human or Human vs. AI
Version 1.5.1
Description A game for two players: Who gets first four in a row (horizontal, vertical or diagonal) wins. As board game published by Milton Bradley, designed by Howard Wexler and Ned Strongin.
Depends R (>= 3.0.0)
License GPL-3
Encoding UTF-8
LazyData true
RoxygenNote 6.1.0
Imports graphics, grDevices
NeedsCompilation no
Author Matthias Speidel [aut, cre]
Maintainer Matthias Speidel <matthias.speidel@googlemail.com>
Repository CRAN
Date/Publication 2018-10-24 15:10:02 UTC

R topics documented:

AImove .................................................. 2
clicking .................................................. 2
clickingXY .............................................. 3
fbuttons ................................................. 3
FieldCorrect .......................................... 4
FieldGeneration ...................................... 4
FieldPlot ............................................... 5
FieldWinCheck ........................................ 5
FourScores ............................................. 6
getColors .............................................. 7
getPlayerNames ....................................... 7
NewField ............................................... 8
### AImove

**Move of AI**

**Description**

Help-Function for an AI

**Usage**

AImove(field, AIstrength, AIplayernumber)

**Arguments**

- **field**: matrix: the playing field
- **AIstrength**: integer: strength of the AI - number of moves the AI will simulate?
- **AIplayernumber**: integer: 0 or 1: should the AI be player 1 or player 2?

**Value**

the selected row

---

### clicking

**a function**

**Description**

help-function which return the x-axis-value of the mouse when releasing the mouse button.

**Usage**

clicking(buttons, x, y)

**Arguments**

- **buttons**: the mouse buttons input.
- **x**: the x-value of the mouse button.
- **y**: the y-value of the mouse button.

**Value**

a rounded value for the x-coordinate
### clickingXY

**Description**

A function to check the mouse click input by the user.

**Usage**

```r
clickingXY(buttons, x, y)
```

**Arguments**

- **buttons**: the mouse buttons input.
- **x**: the x-value of the mouse button.
- **y**: the y-value of the mouse button.

**Value**

A Vector of the x and y coordinates of the mouse click.

---

### fbbuttons

**Field buttons**

**Description**

A function to show buttons, letting the player(s) decide what to do: show the winning field, play again or exit.

**Usage**

```r
fbbuttons(field, justsub, message, MACuser, rows, columns, AI, AIstrength, AIplayernumber, PlayerNames, PlayerColors)
```

**Arguments**

- **field**: matrix: the field.
- **justsub**: boolean: should only be a subtitle plotted (below the winning field)?
- **message**: character: a message to be plotted.
- **MACuser**: boolean: on some non-mac computers this can be set to FALSE to have mouse-functionality in the graphics device.
- **rows**: integer: how many rows shall the playing field have?
- **columns**: integer: how many columns shall the playing field have?
FieldGeneration

AI
boolean: play against AI?

AIstrength
integer: strength of the AI - number of moves the AI will simulate?

AIplayernumber
integer: 0 or 1: should the AI be player 1 or player 2?

PlayerNames
array of characters: the players’ names.

PlayerColors
vector of characters: the players’ colors.

FieldCorrect
Is the field correct?

Description
help-function that checks whether the field is correct

Usage
FieldCorrect(column, field)

Arguments
- column
  integer: the column chosen by the current player
- field
  matrix: the playing field.

Value
a boolean (TRUE if the given column would be a valid move for the field given).

FieldGeneration
field generation

Description
help-function which generates the playing-field

Usage
FieldGeneration(rows, columns)

Arguments
- rows
  integer: how many rows shall the playing field have?
- columns
  integer: how many columns shall the playing field have?

Value
an empty matrix with rows and columns
**FieldPlot**  

*plot the field*

**Description**

a major-function which plots the current field, and if given a hint, which player has won

**Usage**

FieldPlot(field, message, PlayerColors)

**Arguments**

- **field**: matrix: the playing field
- **message**: character: a message to be plotted.
- **PlayerColors**: vector of characters: the players’ colors.

---

**FieldWinCheck**  

*check for a winner*

**Description**

help-function that checks whether (at least) one of the four possibilities of winning is given

**Usage**

FieldWinCheck(field, player)

**Arguments**

- **field**: matrix: the playing field.
- **player**: integer: the current player.

**Value**

a boolean whether the player has won the match or not
FourScores

Main Function

Description

Function to play FourScores

Usage

FourScores(rows = 6, columns = 7, AI = TRUE, AIstrength = rows * columns, AIplayernumber = 1, MACuser = TRUE, PlayerNames = c("AI", "Human"); getnewnames = FALSE, PlayerColors = c("green", "blue"); getnewcolors = FALSE)

Arguments

rows integer: how many rows shall the playing field have?
columns integer: how many columns shall the playing field have?
AI boolean: play against AI?
AIstrength integer: strength of the AI - number of moves the AI will simulate?
AIplayernumber integer: 0 or 1: should the AI be player 1 or player 2?
MACuser boolean: on some non-mac computers this can be set to FALSE to have mouse-functionality in the graphics device.
PlayerNames array of characters: the players’ names.
getnewnames boolean: should new names be asked for?
PlayerColors vector of characters: the players’ colors.
getnewcolors boolean: should new colors be asked for?

Examples

## Not run:
FourScores(AI = T, AIstrength = 10, MACuser = T, getnewnames = F, getnewcolors = F)

## End(Not run)
getColors

**Description**
A function to get some colors

**Usage**
getColors(PlayerNames, PlayerColors, MACuser)

**Arguments**
- **PlayerNames** array of characters: the players’ names.
- **PlayerColors** vector of characters: the players’ colors.
- **MACuser** boolean: on some non-mac computers this can be set to FALSE to have mouse-functionality in the graphics device.

**Value**
a vector with the updated player colors

getPlayerNames

**Description**
help-function which gets and returns the players’ names

**Usage**
getPlayerNames(PlayerNames, MACuser)

**Arguments**
- **PlayerNames** array of characters: the players’ names.
- **MACuser** boolean: on some non-mac computers this can be set to FALSE to have mouse-functionality in the graphics device.

**Value**
a vector with the player names
### NewField

**Generate a new field**

**Description**

help-function which "throws" the stone into the field and returns the new field

**Usage**

`NewField(field, column, player)`

**Arguments**

- **field**: matrix: the playing field.
- **column**: integer: the column chosen by the current player.
- **player**: integer: the current player.

**Value**

The updated field matrix.

---

### painter

**logo painter**

**Description**

a general help function to plot

**Usage**

`painter(numberMatrix, colorArray)`

**Arguments**

- **numberMatrix**: a matrix with different integers showing which color to pick from the `colorArray`.
- **colorArray**: a character array with different names of colors to be used by the painter.
**plotlogo**

---

**Description**

plot the "different purpose" logo

**Usage**

plotlogo()

---

**resample**

---

**Description**

resampling function

**Usage**

resample(x, ...)

**Arguments**

- x: a vector
- ...: other parameters

**Value**

a vector

**References**

Help function from `?sample` to overcome the "sample(ret, size = 1)" problem for `length(ret) == 1"
typing  

---

**Description**
help-function which returns, the key on the keyboard which is being typed

**Usage**
typing(key)

**Arguments**
key  a keyboard input.

**Value**
the key pressed.
Index

AImove, 2
clicking, 2
clickingXY, 3
fbuttons, 3
FieldCorrect, 4
FieldGeneration, 4
FieldPlot, 5
FieldWinCheck, 5
FourScores, 6
getColors, 7
getPlayerNames, 7

NewField, 8
painter, 8
plotlogo, 9
resample, 9
typing, 10