Package ‘rt3’

December 5, 2016

Type Package
Title Tic-Tac-Toe Package for R
Version 0.1.2
Author Johan Jordaan
Maintainer Johan Jordaan <djordaan@gmail.com>
Description Play the classic game of tic-tac-toe (naughts and crosses).
License MIT + file LICENSE
Encoding UTF-8
LazyData true
RoxygenNote 5.0.1
Suggests testthat
NeedsCompilation no
Repository CRAN
Date/Publication 2016-12-05 23:43:08

R topics documented:

EMPTY ................................................................. 2
firstAvailableMovePlayer ........................................ 2
gameState ......................................................... 3
getMoves ......................................................... 3
makeMove ......................................................... 4
NONE .............................................................. 4
O ................................................................. 5
playGame .......................................................... 5
randomMovePlayer ............................................... 6
rt3 ............................................................... 6
startGame ....................................................... 7
X ................................................................. 8

Index 9
firstAvailableMovePlayer

**EMPTY**

*Constant for the empty square. It's value is the character "_".*

**Description**

It’s value is the character "_".

**Usage**

EMPTY

**Format**

An object of class character of length 1.

---

**firstAvailableMovePlayer**

*Player that always takes the first move in the list of valid moves.*

**Description**

Internally this player calls `getMoves` and then picks the first entry in the list of moves. A player is a function that takes a game state as input and returns a valid move index.

**Usage**

firstAvailableMovePlayer(gameState)

**Arguments**

`gameState` The `gameState` that the player should act on.

**Value**

`moveIndex` Index to a valid move as returned by the `getMoves` function.

**Examples**

```r
gameState <- startGame()
move <- firstAvailableMovePlayer(gameState)
```
**gameState**

The game state is represented by a list of 8 values.

**Description**

- **board** The board's state represented by a list. It contains a list of X's, O's and EMPTY's. It's initially filled by EMPTY's.
- **currentPlayer** The player who needs to make the next move. This either X or O.
- **startingPlayer** The player who was the first player to move in this game state. This either X or O.
- **moves** The list of moves made by players to get to this game state. This initially filled with 0's.
- **movesP** The player turn list. It contains a list of alternating X's and O's.
- **numMoves** Number of moves made to get to this game state.
- **isDone** This indicates whether this is a final game state. It is final if either X or O has won if there is no winner: NONE.
- **winner** If there is a winner in this game state the value is either X or O. If the game state is a draw or the game is not finished the value is NONE.

**Usage**

gameState

**Format**

An object of class list of length 8.

**getMoves**

Get the list of valid move from the game state.

**Description**

Get the list of valid move from the game state.

**Usage**

getMoves(gameState)

**Arguments**

- **gameState** The gameState for which moves must be calculated.

**Value**

- validMoves An array (["integer"]) of valid moves based on the provided game state.
Examples

gameState <- startGame()
validMoves <- getMoves(gameState)

makeMove

Apply the move to the current game state and produce a new game state.

Description

Apply the move to the current game state and produce a new game state.

Usage

makeMove(gameState, move)

Arguments

gameState The gameState to apply the move to.
move The move to be applied to the game state.

Value

gameState The game state after applying the move to the game state.

Examples

gameState <- startGame()
gameState <- makeMove(gameState,1)

NONE

Constant for no winner. It's value is the character "_".

Description

It's value is the character "_".

Usage

NONE

Format

An object of class character of length 1.
**Constant for the O player.**

It’s value is the character "O".

**Usage**

0

**Format**

An object of class character of length 1.

---

**playGame**

Play a game of Tic-Tac-Toe using the two provided strategies.

**Description**

Play a game of Tic-Tac-Toe using the two provided strategies.

**Usage**

playGame(px, po)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>px</td>
<td>The X player strategy.</td>
</tr>
<tr>
<td>po</td>
<td>The O player strategy.</td>
</tr>
</tbody>
</table>

**Value**

gameState The final gameState after playing a full game.

**Examples**

px <- firstAvailableMovePlayer
py <- randomMovePlayer
finalGameState <- playGame(px, py)
randomMovePlayer  
*Player that picks a random move*

**Description**

Internally this player calls `getMoves` and then picks an entry in the list of moves at random. A player is a function that takes a game state as input and returns a valid move index.

**Usage**

```r
randomMovePlayer(gameState)
```

**Arguments**

- `gameState` The `gameState` that the player should act on.

**Value**

- `moveIndex` Index to a valid move as returned by the `getMoves` function.

**Examples**

```r
gameState <- startGame()
move <- randomMovePlayer(gameState)
```

---

**rt3**

*rt3: A Package for Playing Tic-Tac-Toe in R.*

**Description**

The `rt3` package provides functions to allow a user to simulate tic-tac-toe games. It provides a convenient `gameState` object as well as simple interface for developing new types of players.

**Main Function**

- `playGame` Play a game of tic-tac-toe.

**Structures**

- `gameState` A tic-tac-toe game state.

**Constants**

- `X` The X player.
- `O` The O player.
- `EMPTY` The EMPTY constant. Used to indicate an empty board position.
- `NONE` The NONE constant. Used to indicate a draw.
Support Functions

These functions are used by the playGame function. They will also be useful in building game decision trees for more complex players.

- **startGame** Create a new tic-tac-toe game state.
- **getMoves** Get the current set of valid moves for a given game state
- **makeMove** Apply a move to the given game state and return the resulting game state

Built-In Player Functions

- **randomMovePlayer** A player that plays random valid moves
- **firstAvailableMovePlayer** A player that always plays the first move available

References


---

**startGame**

**Start a new game**

---

**Description**

This function starts a new game. It randomly assigns a starting player and returns a new game state object.

**Usage**

```r
startGame()
```

**Value**

- *gameState* A new `gameState`

**Examples**

```r
gameState <- startGame()
```
$X$

*Constant for the X player.*

**Description**

It's value is the character "O".

**Usage**

$x$

**Format**

An object of class `character` of length 1.
Index

*Topic datasets
   EMPTY, 2
   gameState, 3
   NONE, 4
   0, 5
   X, 8

EMPTY, 2, 3, 6
firstAvailableMovePlayer, 2, 7
getMoves, 2, 3, 6, 7
makeMove, 4, 7
NONE, 3, 4, 6
0, 3, 5, 6
playGame, 5, 6, 7
randomMovePlayer, 6, 7
rt3, 6
rt3-package (rt3), 6
startGame, 7, 7
X, 3, 6, 8