

Assignment 4

Generated by Doxygen 1.8.13

Contents

| | | |
|----------|--|----------|
| 1 | Class Index | 1 |
| 1.1 | Class List | 1 |
| 2 | File Index | 3 |
| 2.1 | File List | 3 |
| 3 | Class Documentation | 5 |
| 3.1 | src.BoardT Class Reference | 5 |
| 3.1.1 | Detailed Description | 5 |
| 3.1.2 | Constructor & Destructor Documentation | 6 |
| 3.1.2.1 | BoardT() | 6 |
| 3.1.3 | Member Function Documentation | 6 |
| 3.1.3.1 | canMove() | 6 |
| 3.1.3.2 | getBoard() | 6 |
| 3.1.3.3 | getScore() | 7 |
| 3.1.3.4 | getStatus() | 7 |
| 3.1.3.5 | hasWon() | 7 |
| 3.1.3.6 | move() | 7 |
| 3.2 | src.Demo Class Reference | 8 |
| 3.2.1 | Detailed Description | 8 |
| 3.3 | src.DirectionT Enum Reference | 8 |
| 3.3.1 | Detailed Description | 8 |
| 3.4 | src.GameController Class Reference | 9 |
| 3.4.1 | Detailed Description | 9 |

| | | |
|--------------|--|-----------|
| 3.4.2 | Member Function Documentation | 10 |
| 3.4.2.1 | canMove() | 10 |
| 3.4.2.2 | checkWin() | 10 |
| 3.4.2.3 | getInstance() | 10 |
| 3.4.2.4 | getStatus() | 11 |
| 3.4.2.5 | initializeGame() | 11 |
| 3.4.2.6 | move() | 11 |
| 3.4.2.7 | readInput() | 12 |
| 3.4.2.8 | runGame() | 12 |
| 3.5 | src.UserInterface Class Reference | 12 |
| 3.5.1 | Detailed Description | 13 |
| 3.5.2 | Member Function Documentation | 13 |
| 3.5.2.1 | printBoard() | 13 |
| 4 | File Documentation | 15 |
| 4.1 | src/BoardT.java File Reference | 15 |
| 4.1.1 | Detailed Description | 15 |
| 4.2 | src/Demo.java File Reference | 15 |
| 4.2.1 | Detailed Description | 16 |
| 4.3 | src/DirectionT.java File Reference | 16 |
| 4.3.1 | Detailed Description | 16 |
| 4.4 | src/GameController.java File Reference | 16 |
| 4.4.1 | Detailed Description | 17 |
| 4.5 | src/UserInterface.java File Reference | 17 |
| 4.5.1 | Detailed Description | 17 |
| Index | | 19 |

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

| | | |
|------------------------------------|---|----|
| src.BoardT | An abstract data type for the game state of 2048 | 5 |
| src.Demo | A class that the user uses to play the game | 8 |
| src.DirectionT | An enum class for representing the 4 basic directions and printing related messages | 8 |
| src.GameController | An abstract object for dealing with the user input and playing the game | 9 |
| src.UserInterface | An abstract object for representing the game board and printing related messages | 12 |

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

| | | |
|--|---|----|
| src/ BoardT.java | Contains the Abstract Data Type for creating and operating with BoardT objects | 15 |
| src/ Demo.java | Initializes the Board and UserInterface, and runs the game from GameController | 15 |
| src/ DirectionT.java | Contains the enum for specifying the direction that that the game will be moved in | 16 |
| src/ GameController.java | Contains the GameController for initializing and playing the game | 16 |
| src/ UserInterface.java | Contains various messages to be printed during the game, as well as a method for printing the board of the game | 17 |

Chapter 3

Class Documentation

3.1 src.BoardT Class Reference

An abstract data type for the game state of 2048.

Public Member Functions

- **BoardT** (int gameSize, int gameNumber)
*Constructs the **BoardT** object.*
- ArrayList< ArrayList< Integer > > **getBoard** ()
*Gets the board from the **BoardT** object.*
- int **getScore** ()
Gets the current score of the game.
- boolean **getStatus** ()
Gets the current status of the game.
- void **addCell** ()
Adds a value at an empty position on the board.
- boolean **canMove** (**DirectionT** direction)
Determines if the board can be moved in a certain direction.
- boolean **hasWon** ()
Determines whether the game has been won yet.
- void **move** (**DirectionT** direction)
Moves the board in a given direction.

Protected Member Functions

- void **setBoard** (ArrayList< ArrayList< Integer > > board)

3.1.1 Detailed Description

An abstract data type for the game state of 2048.

The game is represented by the board and properties for that board, namely score and status

3.1.2 Constructor & Destructor Documentation

3.1.2.1 BoardT()

```
src.BoardT.BoardT (
    int gameSize,
    int gameNumber )
```

Constructs the [BoardT](#) object.

Parameters

| | |
|-------------------|---|
| <i>gameSize</i> | The square board dimensions of the game |
| <i>gameNumber</i> | The base game number to be played with |

3.1.3 Member Function Documentation

3.1.3.1 canMove()

```
boolean src.BoardT.canMove (
    DirectionT direction )
```

Determines if the board can be moved in a certain direction.

Parameters

| | |
|------------------|--|
| <i>direction</i> | The direction the board is to be moved in. |
|------------------|--|

Returns

Whether the board can be moved in the given direction.

3.1.3.2 getBoard()

```
ArrayList<ArrayList<Integer> > src.BoardT.getBoard ( )
```

Gets the board from the [BoardT](#) object.

Returns

The board of the game.

3.1.3.3 getScore()

```
int src.BoardT.getScore ( )
```

Gets the current score of the game.

Returns

The score of the game.

3.1.3.4 getStatus()

```
boolean src.BoardT.getStatus ( )
```

Gets the current status of the game.

Returns

The status of the game.

3.1.3.5 hasWon()

```
boolean src.BoardT.hasWon ( )
```

Determines whether the game has been won yet.

Returns

The win status of the game.

3.1.3.6 move()

```
void src.BoardT.move (
    DirectionT direction )
```

Moves the board in a given direction.

Parameters

| | |
|------------------|--|
| <i>direction</i> | The direction that the game is to be moved in. |
|------------------|--|

The documentation for this class was generated from the following file:

- [src/BoardT.java](#)

3.2 src.Demo Class Reference

A class that the user uses to play the game.

Static Public Member Functions

- static void **main** (String[] args)

3.2.1 Detailed Description

A class that the user uses to play the game.

Initializes the [BoardT](#) object and calls instances of the [UserInterface](#) and [GameController](#).

The documentation for this class was generated from the following file:

- [src/Demo.java](#)

3.3 src.DirectionT Enum Reference

An enum class for representing the 4 basic directions and printing related messages.

Public Attributes

- **Up**
- **Down**
- **Right**
- **Left**

3.3.1 Detailed Description

An enum class for representing the 4 basic directions and printing related messages.

The specific directions are up, down, right and, left.

The documentation for this enum was generated from the following file:

- [src/DirectionT.java](#)

3.4 src.GameController Class Reference

An abstract object for dealing with the user input and playing the game.

Public Member Functions

- void [initializeGame](#) (int size, int number)
Initializes the game.
- String [readInput](#) ()
Get string input.
- void [addCell](#) ()
Adds a cell to the model object (game board).
- void [move](#) ([DirectionT](#) direction)
Slides board in given direction.
- boolean [getStatus](#) ()
Determines the game's status.
- void [displayWelcomeMessage](#) ()
Updates the view module to display a welcome message.
- void [displayBoard](#) ()
Updates the view module to display the board.
- void [displayEnding](#) ()
Updates the view module to display an ending message.
- void [displayControls](#) ()
Updates the view module to display the controls for the game.
- void [displayScore](#) ()
Updates the view module to display the score of the game.
- void [displayGameOver](#) ()
Updates the view module to display the game over message.
- void [displayGameWon](#) ()
Updates the view module to display the win message.
- boolean [checkWin](#) ()
Determines whether the game has been won.
- boolean [canMove](#) ([DirectionT](#) direction)
Determines if the model module (board) can be moved in the given direction.
- void [runGame](#) ()
Runs the game.

Static Public Member Functions

- static [GameController getInstance](#) ([BoardT](#) model, [UserInterface](#) view)
Public static method for obtaining a single instance.

3.4.1 Detailed Description

An abstract object for dealing with the user input and playing the game.

The game leverages and combines the functionalities of [BoardT](#) and [UserInterface](#).

3.4.2 Member Function Documentation

3.4.2.1 canMove()

```
boolean src.GameController.canMove (
    DirectionT direction )
```

Determines if the model module (board) can be moved in the given direction.

Parameters

| | |
|------------------|--|
| <i>direction</i> | The direction in which to determine whether the module can be moved. |
|------------------|--|

Returns

True if the board can be moved, or False when it can not.

3.4.2.2 checkWin()

```
boolean src.GameController.checkWin ( )
```

Determines whether the game has been won.

Returns

True when the game has been won, and false if it has not.

3.4.2.3 getInstance()

```
static GameController src.GameController.getInstance (
    BoardT model,
    UserInterface view ) [static]
```

Public static method for obtaining a single instance.

Parameters

| | |
|--------------|--|
| <i>model</i> | The game model. |
| <i>view</i> | The UserInterface view |

Returns

The single [GameController](#) object.

3.4.2.4 getStatus()

```
boolean src.GameController.getStatus ( )
```

Determines the game's status.

Returns

The status of the game.

3.4.2.5 initializeGame()

```
void src.GameController.initializeGame (
    int size,
    int number )
```

Initializes the game.

Parameters

| | |
|---------------|------------------------|
| <i>size</i> | The size of the board |
| <i>number</i> | The game's base number |

3.4.2.6 move()

```
void src.GameController.move (
    DirectionT direction )
```

Slides board in given direction.

Parameters

| | |
|------------------|----------------------------------|
| <i>direction</i> | The direction to move the board. |
|------------------|----------------------------------|

3.4.2.7 readInput()

```
String src.GameController.readInput ( )
```

Get string input.

Returns

The input.

3.4.2.8 runGame()

```
void src.GameController.runGame ( )
```

Runs the game.

Checks for multiple forms of conditions and inputs and makes decisions based on them.

The documentation for this class was generated from the following file:

- [src/GameController.java](#)

3.5 src.UserInterface Class Reference

An abstract object for representing the game board and printing related messages.

Public Member Functions

- void [printWelcomeMessage](#) ()
Displays a welcome message.
- void [printGameControlsPrompt](#) ()
Displays a prompt showing the controls of the game.
- void [printGameOver](#) ()
Displays a prompt showing the game over message.
- void [printWin](#) ()
Displays a prompt showing the game win message.
- void [printEndingMessage](#) ()
Displays an ending message after player chooses to exit the game.

Static Public Member Functions

- static [UserInterface](#) [getInstance](#) ()
- static void [printScore](#) ([BoardT](#) model)
Displays a prompt showing the current score of the game.
- static void [printBoard](#) ([BoardT](#) model)
Displays the board on the screen.

3.5.1 Detailed Description

An abstract object for representing the game board and printing related messages.

Includes methods that prints useful messages pertaining to the game board and a size adapting board printing method.

3.5.2 Member Function Documentation

3.5.2.1 printBoard()

```
static void src.UserInterface.printBoard (  
    BoardT model ) [static]
```

Displays the board on the screen.

Parameters

| | |
|--------------|----------------|
| <i>model</i> | The game board |
|--------------|----------------|

The documentation for this class was generated from the following file:

- [src/UserInterface.java](#)

Chapter 4

File Documentation

4.1 src/BoardT.java File Reference

Contains the Abstract Data Type for creating and operating with BoardT objects.

Classes

- class [src.BoardT](#)

An abstract data type for the game state of 2048.

4.1.1 Detailed Description

Contains the Abstract Data Type for creating and operating with BoardT objects.

Author

Mohammad Omar Zahir - zahirm1

Date

April 12, 2021

4.2 src/Demo.java File Reference

Initializes the Board and UserInterface, and runs the game from GameController.

Classes

- class [src.Demo](#)

A class that the user uses to play the game.

4.2.1 Detailed Description

Initializes the Board and UserInterface, and runs the game from GameController.

Author

Mohammad Omar Zahir - zahirm1

Date

April 12, 2021

4.3 src/DirectionT.java File Reference

Contains the enum for specifying the direction that that the game will be moved in.

Classes

- enum [src.DirectionT](#)

An enum class for representing the 4 basic directions and printing related messages.

4.3.1 Detailed Description

Contains the enum for specifying the direction that that the game will be moved in.

Author

Mohammad Omar Zahir - zahirm1

Date

April 12, 2021

4.4 src/GameController.java File Reference

Contains the GameController for initializing and playing the game.

Classes

- class [src.GameController](#)

An abstract object for dealing with the user input and playing the game.

4.4.1 Detailed Description

Contains the GameController for initializing and playing the game.

Author

Mohammad Omar Zahir - zahirm1

Date

April 12, 2021

4.5 src/UserInterface.java File Reference

Contains various messages to be printed during the game, as well as a method for printing the board of the game.

Classes

- class [src.UserInterface](#)

An abstract object for representing the game board and printing related messages.

4.5.1 Detailed Description

Contains various messages to be printed during the game, as well as a method for printing the board of the game.

Author

Mohammad Omar Zahir - zahirm1

Date

April 12, 2021

Index

BoardT
 src::BoardT, [6](#)

canMove
 src::BoardT, [6](#)
 src::GameController, [10](#)

checkWin
 src::GameController, [10](#)

getBoard
 src::BoardT, [6](#)

getInstance
 src::GameController, [10](#)

getScore
 src::BoardT, [6](#)

getStatus
 src::BoardT, [7](#)
 src::GameController, [11](#)

hasWon
 src::BoardT, [7](#)

initializeGame
 src::GameController, [11](#)

move
 src::BoardT, [7](#)
 src::GameController, [11](#)

printBoard
 src::UserInterface, [13](#)

readInput
 src::GameController, [11](#)

runGame
 src::GameController, [12](#)

src.BoardT, [5](#)
src.Demo, [8](#)
src.DirectionT, [8](#)
src.GameController, [9](#)
src.UserInterface, [12](#)
src/BoardT.java, [15](#)
src/Demo.java, [15](#)
src/DirectionT.java, [16](#)
src/GameController.java, [16](#)
src/UserInterface.java, [17](#)
src::BoardT
 BoardT, [6](#)
 canMove, [6](#)
 getBoard, [6](#)
 getScore, [6](#)
 getStatus, [7](#)
 hasWon, [7](#)
 move, [7](#)
src::GameController
 canMove, [10](#)
 checkWin, [10](#)
 getInstance, [10](#)
 getStatus, [11](#)
 initializeGame, [11](#)
 move, [11](#)
 readInput, [11](#)
 runGame, [12](#)
src::UserInterface
 printBoard, [13](#)