

## Assignment 4: Game Of Dots

Generated by Doxygen 1.8.13



# Contents

<b>1</b>	<b>Class Index</b>	<b>1</b>
1.1	Class List . . . . .	1
<b>2</b>	<b>File Index</b>	<b>3</b>
2.1	File List . . . . .	3
<b>3</b>	<b>Class Documentation</b>	<b>5</b>
3.1	AllTests Class Reference . . . . .	5
3.2	src.CellT Enum Reference . . . . .	5
3.2.1	Detailed Description . . . . .	5
3.2.2	Member Function Documentation . . . . .	5
3.2.2.1	getRandomCell() . . . . .	6
3.3	src.Controller Class Reference . . . . .	6
3.3.1	Detailed Description . . . . .	6
3.3.2	Constructor & Destructor Documentation . . . . .	7
3.3.2.1	Controller() . . . . .	7
3.3.3	Member Function Documentation . . . . .	7
3.3.3.1	execute_move() . . . . .	7
3.3.3.2	game_over() . . . . .	7
3.3.3.3	get_game_boardT() . . . . .	8
3.3.3.4	get_game_moves() . . . . .	8
3.3.3.5	get_game_obj() . . . . .	8
3.3.3.6	get_game_score() . . . . .	9
3.3.3.7	is_win() . . . . .	9

3.3.3.8	updateView()	9
3.4	src.GameBoardT Class Reference	9
3.4.1	Detailed Description	10
3.4.2	Constructor & Destructor Documentation	10
3.4.2.1	GameBoardT()	10
3.4.3	Member Function Documentation	10
3.4.3.1	game_over()	11
3.4.3.2	get_boardT()	11
3.4.3.3	get_moves()	11
3.4.3.4	get_obj()	11
3.4.3.5	get_score()	12
3.4.3.6	initialize_board()	12
3.4.3.7	is_win()	12
3.4.3.8	move()	12
3.5	src.TestCellT Class Reference	13
3.6	src.TestController Class Reference	13
3.7	src.TestGameBoardT Class Reference	14
3.8	src.View Class Reference	14
3.8.1	Detailed Description	14
3.8.2	Member Function Documentation	14
3.8.2.1	print_board()	14
3.8.2.2	print_info()	15
<b>4</b>	<b>File Documentation</b>	<b>17</b>
4.1	src/CellT.java File Reference	17
4.1.1	Detailed Description	17
4.2	src/Controller.java File Reference	17
4.2.1	Detailed Description	18
4.3	src/GameBoardT.java File Reference	18
4.3.1	Detailed Description	18
4.4	src/TestCellT.java File Reference	18
4.4.1	Detailed Description	19
4.5	src/TestGameBoardT.java File Reference	19
4.5.1	Detailed Description	19
4.6	src/View.java File Reference	19
4.6.1	Detailed Description	19
<b>Index</b>		<b>21</b>

# Chapter 1

## Class Index

### 1.1 Class List

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

<a href="#">AllTests</a>	5
<a href="#">src.CellT</a>	
This class represents <a href="#">CellT</a>	5
<a href="#">src.Controller</a>	
This class outputs the controller to update model and view	6
<a href="#">src.GameBoardT</a>	
Class represents the <a href="#">GameBoardT</a> module to play Dots Game	9
<a href="#">src.TestCellT</a>	13
<a href="#">src.TestController</a>	13
<a href="#">src.TestGameBoardT</a>	14
<a href="#">src.View</a>	
This class outputs the board and game details to the user	14



## Chapter 2

# File Index

### 2.1 File List

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

src/ <a href="#">CellT.java</a>	
Enum Java class for CellT . . . . .	17
src/ <a href="#">Controller.java</a>	
Controller class for model and view modules . . . . .	17
src/ <a href="#">GameBoardT.java</a>	
GameBoardT ADT module that uses CellT module . . . . .	18
src/ <a href="#">TestCellT.java</a>	
Test Driver for CellT module . . . . .	18
src/ <a href="#">TestGameBoardT.java</a>	
Test Driver for GameBoardT module . . . . .	19
src/ <a href="#">View.java</a>	
View module that uses CellT and GameBoardT modules . . . . .	19





## Chapter 3

# Class Documentation

### 3.1 AllTests Class Reference

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

- `src/AllTests.java`

### 3.2 `src.CellT` Enum Reference

This class represents `CellT`.

#### Static Public Member Functions

- static `CellT getRandomCell ()`  
*method that returns a random `CellT` type*

#### Public Attributes

- `R`
- `O`
- `G`

#### 3.2.1 Detailed Description

This class represents `CellT`.

This is an enum class for the various `CellT` types

#### 3.2.2 Member Function Documentation

### 3.2.2.1 getRandomCell()

```
static CellT src.CellT.getRandomCell ( ) [static]
```

method that returns a random [CellT](#) type

Method that returns either of the four [CellT](#) colors

#### Returns

Returns a random [CellT](#) value

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

- [src/CellT.java](#)

## 3.3 src.Controller Class Reference

This class outputs the controller to update model and view.

### Public Member Functions

- [Controller](#) ([GameBoardT](#) gameboard, [View](#) view)  
*Constructor that that initializes for the controller.*
- [CellT](#) [][] [get\\_game\\_boardT](#) ()  
*Getter method used to retrieve boardT grid.*
- [ArrayList](#) [get\\_game\\_obj](#) ()  
*Getter method used to retrieve objectives.*
- int [get\\_game\\_score](#) ()  
*Getter method used to retrieve score.*
- int [get\\_game\\_moves](#) ()  
*Getter method used to retrieve number of moves.*
- void [execute\\_move](#) (int[] path)  
*[Controller](#) Method that updates the boardT after a valid move.*
- void [updateView](#) ()  
*[Controller](#) method that updates the view output for the user.*
- boolean [game\\_over](#) ()  
*Checks if game is over when ran out of available moves.*
- boolean [is\\_win](#) ()  
*Checks if user has won the game.*

### 3.3.1 Detailed Description

This class outputs the controller to update model and view.

[Controller](#) of the MVC design pattern

### 3.3.2 Constructor & Destructor Documentation

#### 3.3.2.1 Controller()

```
src.Controller.Controller (
    GameBoardT gameboard,
    View view )
```

Constructor that that initializes for the controller.

Initializes the model(gameboard) and view modules

##### Parameters

<i>gameboard</i>	represents a GameboardT object
<i>view</i>	represents a <a href="#">View</a> object

### 3.3.3 Member Function Documentation

#### 3.3.3.1 execute\_move()

```
void src.Controller.execute_move (
    int path[ ][ ] )
```

[Controller](#) Method that updates the boardT after a valid move.

Mutator method that updates the boardT with a valid path from user

##### Parameters

<i>path</i>	represents the 2D integer array of the path chosen by the client
-------------	--

##### Exceptions

<i>NullPointerException, IndexOutOfBoundsException, IllegalArgumentException</i>	
--	--

#### 3.3.3.2 game\_over()

```
boolean src.Controller.game_over ( )
```

Checks if game is over when ran out of available moves.

Outputs a boolean signifying if available moves is  $\leq 0$

#### Returns

returns True if available moves is less than or equal to 0

#### 3.3.3.3 `get_game_boardT()`

```
CellT [][] src.Controller.get_game_boardT ( )
```

Getter method used to retrieve boardT grid.

getter method to return the boardT state variable

#### Returns

Returns a 2D array of `CellT` representing the boardT

#### 3.3.3.4 `get_game_moves()`

```
int src.Controller.get_game_moves ( )
```

Getter method used to retrieve number of moves.

Gets the number of moves remaining anytime during the game

#### Returns

Returns an integer containing the number of moves remaining

#### 3.3.3.5 `get_game_obj()`

```
ArrayList src.Controller.get_game_obj ( )
```

Getter method used to retrieve objectives.

getter method to return the objectives state variable

#### Returns

Returns an arraylist containing objective parameters

**3.3.3.6 get\_game\_score()**

```
int src.Controller.get_game_score ( )
```

Getter method used to retrieve score.

getter method to return the score state variable

**Returns**

Returns an integer containing the score

**3.3.3.7 is\_win()**

```
boolean src.Controller.is_win ( )
```

Checks if user has won the game.

Outputs a boolean if user has met the objective

**Returns**

returns True if objective has reached or gone below 0

**3.3.3.8 updateView()**

```
void src.Controller.updateView ( )
```

[Controller](#) method that updates the view output for the user.

Outputs the state variables for the game by printing to the console

**Exceptions**

<i>NullPointerException, IndexOutOfBoundsException, IllegalArgumentException</i>
--

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

- [src/Controller.java](#)

**3.4 src.GameBoardT Class Reference**

class represents the [GameBoardT](#) module to play Dots Game

## Public Member Functions

- [GameBoardT](#) ()  
*Constructor that initializes the [GameBoardT](#) object.*
- void [initialize\\_board](#) ()  
*Initializes the GameBoard for the state variables.*
- [CellT](#) [][] [get\\_boardT](#) ()  
*Getter method used to retrieve boardT grid.*
- ArrayList [get\\_obj](#) ()  
*Getter method used to retrieve objectives.*
- int [get\\_score](#) ()  
*Getter method used to retrieve score.*
- int [get\\_moves](#) ()  
*Getter method used to retrieve number of moves.*
- void [move](#) (int[] [] path)  
*Method that updates the boardT after a valid move.*
- boolean [game\\_over](#) ()  
*Checks if game is over when ran out of available moves.*
- boolean [is\\_win](#) ()  
*Checks if user has won the game.*

### 3.4.1 Detailed Description

class represents the [GameBoardT](#) module to play Dots Game

The model of MVC with various operations on a 6X6 grid

### 3.4.2 Constructor & Destructor Documentation

#### 3.4.2.1 [GameBoardT](#)()

```
src.GameBoardT.GameBoardT ( )
```

Constructor that initializes the [GameBoardT](#) object.

[GameBoardT](#) constructor takes no parameters to create [GameBoardT](#) object

### 3.4.3 Member Function Documentation

#### 3.4.3.1 game\_over()

```
boolean src.GameBoardT.game_over ( )
```

Checks if game is over when ran out of available moves.

Outputs a boolean signifying if available moves is  $\leq 0$

##### Returns

returns True if available moves is less than or equal to 0

#### 3.4.3.2 get\_boardT()

```
CellT [][] src.GameBoardT.get_boardT ( )
```

Getter method used to retrieve boardT grid.

getter method to return the boardT state variable

##### Returns

Returns a 2D array of [CellT](#) representing the boardT

#### 3.4.3.3 get\_moves()

```
int src.GameBoardT.get_moves ( )
```

Getter method used to retrieve number of moves.

Gets the number of moves remaining anytime during the game

##### Returns

Returns an integer containing the number of moves remaining

#### 3.4.3.4 get\_obj()

```
ArrayList src.GameBoardT.get_obj ( )
```

Getter method used to retrieve objectives.

getter method to return the objectives state variable

##### Returns

Returns an arraylist containing objective parameters

#### 3.4.3.5 get\_score()

```
int src.GameBoardT.get_score ( )
```

Getter method used to retrieve score.

getter method to return the score state variable

##### Returns

Returns an integer containing the score

#### 3.4.3.6 initialize\_board()

```
void src.GameBoardT.initialize_board ( )
```

Initializes the GameBoard for the state variables.

Takes no parameters. Assigns the default properties for the board

#### 3.4.3.7 is\_win()

```
boolean src.GameBoardT.is_win ( )
```

Checks if user has won the game.

Outputs a boolean if user has met the objective

##### Returns

returns True if objective has reached or gone below 0

#### 3.4.3.8 move()

```
void src.GameBoardT.move (
    intpath [ ][ ] )
```

Method that updates the boardT after a valid move.

Mutator method that updates the boardT with a valid path from user

##### Parameters

<i>path</i>	represents the 2D integer array of the path chosen by the client
-------------	--



## Exceptions

<i>NullPointerException, IndexOutOfBoundsException, IllegalArgumentException</i>
--

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

- [src/GameBoardT.java](#)

## 3.5 src.TestCellT Class Reference

### Public Member Functions

- void **setUp** () throws Exception
- void **tearDown** () throws Exception
- void **test\_getRandomCell** ()

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

- [src/TestCellT.java](#)

## 3.6 src.TestController Class Reference

### Public Member Functions

- void **setUp** () throws Exception
- void **tearDown** () throws Exception
- void **test\_get\_boardT** ()
- void **test\_get\_obj** ()
- void **test\_get\_score** ()
- void **test\_get\_moves** ()
- void **test\_game\_over** ()
- void **test\_is\_win** ()
- void **test\_move** ()

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

- [src/TestController.java](#)

## 3.7 src.TestGameBoardT Class Reference

### Public Member Functions

- void **setUp** () throws Exception
- void **tearDown** () throws Exception
- void **test\_get\_boardT** ()
- void **test\_get\_obj** ()
- void **test\_get\_score** ()
- void **test\_get\_moves** ()
- void **test\_game\_over** ()
- void **test\_is\_win** ()
- void **test\_move** ()

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

- [src/TestGameBoardT.java](#)

## 3.8 src.View Class Reference

This class outputs the board and game details to the user.

### Public Member Functions

- void [print\\_board](#) ([CellT](#)[][] boardT)  
*Method used to show the board to the user.*
- void [print\\_info](#) ([CellT](#) objective\_type, int objective\_goal, int score, int moves)  
*Method used to show the users status in the game.*

### 3.8.1 Detailed Description

This class outputs the board and game details to the user.

This is the view portion of the MVC design pattern

### 3.8.2 Member Function Documentation

#### 3.8.2.1 print\_board()

```
void src.View.print_board (  
    CellT boardT[ ][ ] )
```

Method used to show the board to the user.

Prints the board to the console to view

**Parameters**

<i>boardT</i>	is a 2D sequence of <a href="#">CellT</a> that represents the game board
---------------	--

**3.8.2.2 print\_info()**

```
void src.View.print_info (
    CellT objective_type,
    int objective_goal,
    int score,
    int moves )
```

Method used to show the users status in the game.

Prints the current objectives, moves, and score to the console

**Parameters**

<i>objective_type</i>	represents the <a href="#">CellT</a> objective
<i>objective_goal</i>	represents the target goal for <a href="#">CellT</a> objective
<i>score</i>	represents the running total of the users points
<i>moves</i>	represents total available moves for the user

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

- [src/View.java](#)



## Chapter 4

# File Documentation

### 4.1 src/CellT.java File Reference

Enum Java class for CellT.

#### Classes

- enum [src.CellT](#)

*This class represents [CellT](#).*

#### 4.1.1 Detailed Description

Enum Java class for CellT.

#### Author

Anando Zaman

#### Date

March 25,2020

Used for the exported LandTypes

### 4.2 src/Controller.java File Reference

Controller class for model and view modules.

#### Classes

- class [src.Controller](#)

*This class outputs the controller to update model and view.*

### 4.2.1 Detailed Description

Controller class for model and view modules.

**Author**

Anando Zaman

**Date**

March 25,2020

Used to control/update model(GameBoardT) and view modules

## 4.3 src/GameBoardT.java File Reference

GameBoardT ADT module that uses CellT module.

### Classes

- class [src.GameBoardT](#)  
*class represents the [GameBoardT](#) module to play Dots Game*

### 4.3.1 Detailed Description

GameBoardT ADT module that uses CellT module.

**Author**

Anando Zaman

**Date**

March 25,2020

Model of MVC for playing the Dots game on a 6X6 GRID

## 4.4 src/TestCellT.java File Reference

Test Driver for CellT module.

### Classes

- class [src.TestCellT](#)

#### 4.4.1 Detailed Description

Test Driver for CellT module.

Author

Anando Zaman

Date

March 25,2020

### 4.5 src/TestGameBoardT.java File Reference

Test Driver for GameBoardT module.

#### Classes

- class [src.TestGameBoardT](#)

#### 4.5.1 Detailed Description

Test Driver for GameBoardT module.

Test Driver for Controller module.

Author

Anando Zaman

Date

March 25,2020

### 4.6 src/View.java File Reference

View module that uses CellT and GameBoardT modules.

#### Classes

- class [src.View](#)  
*This class outputs the board and game details to the user.*

#### 4.6.1 Detailed Description

View module that uses CellT and GameBoardT modules.

Author

Anando Zaman

Date

March 25,2020

View of MVC for outputting results to the user





# Index

AllTests, [5](#)

Controller  
    src::Controller, [7](#)

execute\_move  
    src::Controller, [7](#)

game\_over  
    src::Controller, [7](#)  
    src::GameBoardT, [10](#)

GameBoardT  
    src::GameBoardT, [10](#)

get\_boardT  
    src::GameBoardT, [11](#)

get\_game\_boardT  
    src::Controller, [8](#)

get\_game\_moves  
    src::Controller, [8](#)

get\_game\_obj  
    src::Controller, [8](#)

get\_game\_score  
    src::Controller, [8](#)

get\_moves  
    src::GameBoardT, [11](#)

get\_obj  
    src::GameBoardT, [11](#)

get\_score  
    src::GameBoardT, [11](#)

getRandomCell  
    src::CellT, [5](#)

initialize\_board  
    src::GameBoardT, [12](#)

is\_win  
    src::Controller, [9](#)  
    src::GameBoardT, [12](#)

move  
    src::GameBoardT, [12](#)

print\_board  
    src::View, [14](#)

print\_info  
    src::View, [15](#)

src.CellT, [5](#)  
src.Controller, [6](#)  
src.GameBoardT, [9](#)  
src.TestCellT, [13](#)  
src.TestController, [13](#)  
src.TestGameBoardT, [14](#)  
src.View, [14](#)  
src/CellT.java, [17](#)  
src/Controller.java, [17](#)  
src/GameBoardT.java, [18](#)  
src/TestCellT.java, [18](#)  
src/TestGameBoardT.java, [19](#)  
src/View.java, [19](#)  
src::CellT  
    getRandomCell, [5](#)  
src::Controller  
    Controller, [7](#)  
    execute\_move, [7](#)  
    game\_over, [7](#)  
    get\_game\_boardT, [8](#)  
    get\_game\_moves, [8](#)  
    get\_game\_obj, [8](#)  
    get\_game\_score, [8](#)  
    is\_win, [9](#)  
    updateView, [9](#)  
src::GameBoardT  
    game\_over, [10](#)  
    GameBoardT, [10](#)  
    get\_boardT, [11](#)  
    get\_moves, [11](#)  
    get\_obj, [11](#)  
    get\_score, [11](#)  
    initialize\_board, [12](#)  
    is\_win, [12](#)  
    move, [12](#)  
src::View  
    print\_board, [14](#)  
    print\_info, [15](#)  
updateView  
    src::Controller, [9](#)