

# The Good The Bad and The Ugly Scrabble Board Game

4.0

Generated by Doxygen 1.8.17



<b>1 Hierarchical Index</b>	<b>1</b>
1.1 Class Hierarchy	1
<b>2 Class Index</b>	<b>3</b>
2.1 Class List	3
<b>3 Class Documentation</b>	<b>5</b>
3.1 scrabble.Board Class Reference	5
3.1.1 Detailed Description	5
3.1.2 Constructor & Destructor Documentation	6
3.1.2.1 Board()	6
3.1.3 Member Function Documentation	6
3.1.3.1 calculateScore()	6
3.1.3.2 checkPlayerHasTiles()	6
3.1.3.3 checkValidMove()	7
3.1.3.4 checkValidPosition()	7
3.1.3.5 getBoardSquares()	7
3.1.3.6 getSquare()	8
3.1.3.7 placeTile()	8
3.1.3.8 placeTiles()	9
3.1.3.9 removeMove()	9
3.1.3.10 resetBoard()	9
3.1.3.11 setWordSquaresNormal()	9
3.1.3.12 toString()	9
3.1.4 Member Data Documentation	10
3.1.4.1 BINGO	10
3.1.4.2 BOARD_SIZE	10
3.2 scrabble.BoardTest Class Reference	10
3.3 scrabble.UserInput.Direction Enum Reference	10
3.3.1 Detailed Description	10
3.4 scrabble.Frame Class Reference	11
3.4.1 Detailed Description	11
3.4.2 Constructor & Destructor Documentation	11
3.4.2.1 Frame()	11
3.4.3 Member Function Documentation	12
3.4.3.1 addTile()	12
3.4.3.2 checkTiles() [1/2]	12
3.4.3.3 checkTiles() [2/2]	12
3.4.3.4 fillFrame()	13
3.4.3.5 getTile() [1/2]	13
3.4.3.6 getTile() [2/2]	13
3.4.3.7 getTiles()	14
3.4.3.8 hasBlank()	14

3.4.3.9 isEmpty()	15
3.4.3.10 removeTile() [1/3]	15
3.4.3.11 removeTile() [2/3]	15
3.4.3.12 removeTile() [3/3]	16
3.4.3.13 removeTiles() [1/2]	16
3.4.3.14 removeTiles() [2/2]	16
3.4.3.15 returnFrame()	17
3.4.3.16 setBlanks()	17
3.4.3.17 setToBlank()	17
3.4.3.18 swapTiles()	17
3.4.3.19 tileValues()	18
3.4.3.20 toString()	18
3.4.4 Member Data Documentation	18
3.4.4.1 FRAME_SIZE	18
3.5 scrabble.exceptions.InvalidBoardException Class Reference	18
3.5.1 Detailed Description	19
3.6 scrabble.exceptions.InvalidFrameException Class Reference	19
3.6.1 Detailed Description	19
3.7 scrabble.exceptions.InvalidInputException Class Reference	19
3.7.1 Detailed Description	20
3.8 scrabble.exceptions.InvalidMoveInfoException Class Reference	20
3.8.1 Detailed Description	20
3.9 scrabble.exceptions.InvalidPlayerNameException Class Reference	20
3.9.1 Detailed Description	21
3.10 scrabble.exceptions.InvalidPlayerScoreException Class Reference	21
3.10.1 Detailed Description	21
3.11 scrabble.exceptions.InvalidPoolException Class Reference	21
3.11.1 Detailed Description	22
3.12 scrabble.exceptions.InvalidScrabbleException Class Reference	22
3.12.1 Detailed Description	22
3.13 scrabble.exceptions.InvalidSquareException Class Reference	22
3.13.1 Detailed Description	23
3.14 scrabble.exceptions.InvalidTileException Class Reference	23
3.14.1 Detailed Description	23
3.15 scrabble.exceptions.InvalidWordException Class Reference	23
3.15.1 Detailed Description	24
3.16 scrabble.Main Class Reference	24
3.17 scrabble.MoveInfo Class Reference	24
3.17.1 Detailed Description	24
3.17.2 Constructor & Destructor Documentation	25
3.17.2.1 MoveInfo()	25
3.17.3 Member Function Documentation	26

3.17.3.1 addAuxiliaryWord()	26
3.17.3.2 getAuxiliaryWords()	26
3.17.3.3 getMoveScore()	26
3.17.3.4 getPlayer()	27
3.17.3.5 getPrimaryWord()	27
3.17.3.6 getRequiredTiles()	27
3.17.3.7 getRequiredTilesPositions()	27
3.17.3.8 setRequiredTiles()	27
3.17.3.9 setScore()	28
3.18 scrabble.MoveInfoTest Class Reference	28
3.19 scrabble.Player Class Reference	28
3.19.1 Detailed Description	29
3.19.2 Constructor & Destructor Documentation	29
3.19.2.1 Player()	29
3.19.3 Member Function Documentation	29
3.19.3.1 decreaseScore()	29
3.19.3.2 getName()	30
3.19.3.3 getPlayerFrame()	30
3.19.3.4 getScore()	30
3.19.3.5 increaseScore()	30
3.19.3.6 playerReset()	31
3.19.3.7 setName()	31
3.19.3.8 toString()	31
3.20 scrabble.Pool Class Reference	32
3.20.1 Detailed Description	32
3.20.2 Constructor & Destructor Documentation	32
3.20.2.1 Pool()	32
3.20.3 Member Function Documentation	32
3.20.3.1 isEmpty()	33
3.20.3.2 poolFill()	33
3.20.3.3 receiveTile()	33
3.20.3.4 removeTile()	33
3.20.3.5 tilesInPool()	34
3.20.3.6 toString()	34
3.21 scrabble.PoolTest Class Reference	34
3.22 scrabble.Scrabble Class Reference	34
3.22.1 Constructor & Destructor Documentation	34
3.22.1.1 Scrabble()	34
3.22.2 Member Function Documentation	35
3.22.2.1 createPlayer()	35
3.22.2.2 dictionaryWords()	35
3.22.2.3 gameOver()	36

3.22.2.4	getBoard()	36
3.22.2.5	getMoveHistory()	36
3.22.2.6	getPlayers()	36
3.22.2.7	getPool()	36
3.22.2.8	isGameOver()	37
3.22.2.9	playerMove()	37
3.23	scrabble.ScrabbleTest Class Reference	37
3.24	scrabble.Square Class Reference	37
3.24.1	Detailed Description	38
3.24.2	Constructor & Destructor Documentation	38
3.24.2.1	Square()	38
3.24.3	Member Function Documentation	38
3.24.3.1	getTile()	38
3.24.3.2	getType()	39
3.24.3.3	isEmpty()	39
3.24.3.4	setEmpty()	39
3.24.3.5	setNormal()	39
3.24.3.6	setTile()	40
3.24.3.7	toString()	40
3.25	scrabble.SquareTest Class Reference	40
3.26	scrabble.Square.SquareType Enum Reference	40
3.26.1	Detailed Description	41
3.27	scrabble.Tile Class Reference	41
3.27.1	Detailed Description	41
3.27.2	Constructor & Destructor Documentation	41
3.27.2.1	Tile()	41
3.27.3	Member Function Documentation	42
3.27.3.1	compareTo()	42
3.27.3.2	equals()	42
3.27.3.3	getCharacter()	43
3.27.3.4	getValue()	43
3.27.3.5	main()	43
3.27.3.6	setCharacter()	43
3.27.3.7	setNull()	44
3.27.3.8	toString()	44
3.28	scrabble.UserInput Class Reference	44
3.28.1	Constructor & Destructor Documentation	45
3.28.1.1	UserInput() [ 1 / 4 ]	45
3.28.1.2	UserInput() [ 2 / 4 ]	45
3.28.1.3	UserInput() [ 3 / 4 ]	45
3.28.1.4	UserInput() [ 4 / 4 ]	46
3.28.2	Member Function Documentation	46

3.28.2.1 getInputType()	46
3.28.2.2 getName()	46
3.28.2.3 getStartPosition()	47
3.28.2.4 getWord()	47
3.28.2.5 getWordDirection()	47
3.28.2.6 parseInput()	47
3.29 scrabble.UserInputTest Class Reference	48
3.30 scrabble.UserInput.UserInputType Enum Reference	48
3.30.1 Detailed Description	48
3.31 scrabble.userInterface.UserInterface Class Reference	48
3.31.1 Member Function Documentation	49
3.31.1.1 main()	49
3.31.1.2 start()	49
3.32 scrabble.Word Class Reference	49
3.32.1 Detailed Description	50
3.32.2 Constructor & Destructor Documentation	50
3.32.2.1 Word()	50
3.32.3 Member Function Documentation	50
3.32.3.1 getDirection()	50
3.32.3.2 getStartPosition()	51
3.32.3.3 getWord()	51
3.32.3.4 toString()	51
3.33 scrabble.WordTest Class Reference	51
<b>Index</b>	<b>53</b>





## Chapter 1

# Hierarchical Index

### 1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

scrabble.Board	5
scrabble.BoardTest	10
Comparable	
scrabble.Tile	41
scrabble.UserInput.Direction	10
scrabble.Frame	11
IllegalArgumentException	
scrabble.exceptions.InvalidBoardException	18
scrabble.exceptions.InvalidFrameException	19
scrabble.exceptions.InvalidInputException	19
scrabble.exceptions.InvalidMoveInfoException	20
scrabble.exceptions.InvalidPlayerNameException	20
scrabble.exceptions.InvalidPlayerScoreException	21
scrabble.exceptions.InvalidScrabbleException	22
scrabble.exceptions.InvalidSquareException	22
scrabble.exceptions.InvalidTileException	23
scrabble.exceptions.InvalidWordException	23
scrabble.Main	24
scrabble.MoveInfo	24
scrabble.MoveInfoTest	28
scrabble.Player	28
scrabble.Pool	32
scrabble.PoolTest	34
scrabble.Scrabble	34
scrabble.ScrabbleTest	37
scrabble.Square	37
scrabble.SquareTest	40
scrabble.Square.SquareType	40
scrabble.UserInput	44
scrabble.UserInputTest	48
scrabble.UserInput.UserInputType	48
scrabble.Word	49
scrabble.WordTest	51
Application	
scrabble.userInterface.UserInterface	48
NoSuchElementException	
scrabble.exceptions.InvalidPoolException	21



## Chapter 2

# Class Index

### 2.1 Class List

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

<a href="#">scrabble.Board</a>	5
<a href="#">scrabble.BoardTest</a>	10
<a href="#">scrabble.UserInput.Direction</a>	10
<a href="#">scrabble.Frame</a>	11
<a href="#">scrabble.exceptions.InvalidBoardException</a>	18
<a href="#">scrabble.exceptions.InvalidFrameException</a>	19
<a href="#">scrabble.exceptions.InvalidInputException</a>	19
<a href="#">scrabble.exceptions.InvalidMoveInfoException</a>	20
<a href="#">scrabble.exceptions.InvalidPlayerNameException</a>	20
<a href="#">scrabble.exceptions.InvalidPlayerScoreException</a>	21
<a href="#">scrabble.exceptions.InvalidPoolException</a>	21
<a href="#">scrabble.exceptions.InvalidScrabbleException</a>	22
<a href="#">scrabble.exceptions.InvalidSquareException</a>	22
<a href="#">scrabble.exceptions.InvalidTileException</a>	23
<a href="#">scrabble.exceptions.InvalidWordException</a>	23
<a href="#">scrabble.Main</a>	24
<a href="#">scrabble.MoveInfo</a>	24
<a href="#">scrabble.MoveInfoTest</a>	28
<a href="#">scrabble.Player</a>	28
<a href="#">scrabble.Pool</a>	32
<a href="#">scrabble.PoolTest</a>	34
<a href="#">scrabble.Scrabble</a>	34
<a href="#">scrabble.ScrabbleTest</a>	37
<a href="#">scrabble.Square</a>	37
<a href="#">scrabble.SquareTest</a>	40
<a href="#">scrabble.Square.SquareType</a>	40
<a href="#">scrabble.Tile</a>	41
<a href="#">scrabble.UserInput</a>	44
<a href="#">scrabble.UserInputTest</a>	48
<a href="#">scrabble.UserInput.UserInputType</a>	48
<a href="#">scrabble.userInterface.UserInterface</a>	48
<a href="#">scrabble.Word</a>	49
<a href="#">scrabble.WordTest</a>	51



## Chapter 3

# Class Documentation

### 3.1 scrabble.Board Class Reference

#### Public Member Functions

- [Board](#) ()
- [Square](#)[][] [getBoardSquares](#) ()
- void [resetBoard](#) ()
- [Square](#) [getSquare](#) (int i, int j)
- String [toString](#) ()
- void [placeTiles](#) ([MoveInfo](#) moveInfo)
- void [setWordSquaresNormal](#) ([Word](#) word)
- void [removeMove](#) ([MoveInfo](#) moveInfo)

#### Static Public Member Functions

- static Boolean [checkValidPosition](#) (int[] position)

#### Static Public Attributes

- static final int [BOARD\\_SIZE](#) = 15
- static final int [BINGO](#) = 50

#### Protected Member Functions

- boolean [checkValidMove](#) ([MoveInfo](#) moveInfo)
- void [placeTile](#) ([Tile](#) tile, int position\_i, int position\_j)
- boolean [checkPlayerHasTiles](#) ([Player](#) player, char[] word)
- int [calculateScore](#) ([MoveInfo](#) moveInfo)

#### 3.1.1 Detailed Description

The [Board](#) Class represents the [Board](#) for the [Scrabble](#) Game as an object

## 3.1.2 Constructor & Destructor Documentation

### 3.1.2.1 Board()

```
scrabble.Board.Board ( )
```

[Board](#) Constructor

## 3.1.3 Member Function Documentation

### 3.1.3.1 calculateScore()

```
int scrabble.Board.calculateScore (
    MoveInfo moveInfo ) [protected]
```

Method to calculate the score of a move

Parameters

<i>moveInfo</i>	Move to calculate the score
-----------------	-----------------------------

Returns

The score of the move

### 3.1.3.2 checkPlayerHasTiles()

```
boolean scrabble.Board.checkPlayerHasTiles (
    Player player,
    char[] word ) [protected]
```

Method to validate if the [Player](#) placing a list of Tiles has each [Tile](#) in his [Frame](#)

Parameters

<i>player</i>	<a href="#">Player</a> to check if they have the necessary Tiles
<i>word</i>	List of Tiles to check

**Returns**

True if the player has the tiles

**3.1.3.3 checkValidMove()**

```
boolean scrabble.Board.checkValidMove (
    MoveInfo moveInfo ) [protected]
```

Method to check that a move from the player is valid

**Parameters**

<i>moveInfo</i>	The move to validate
-----------------	----------------------

**Returns**

true if valid move

**Exceptions**

<i>InvalidMoveInfoException</i>	If the Move is Invalid
---------------------------------	------------------------

**3.1.3.4 checkValidPosition()**

```
static Boolean scrabble.Board.checkValidPosition (
    int[] position ) [static]
```

Method to validate that a position passed in is on the board

**Parameters**

<i>position</i>	Position to check if its on the board
-----------------	---------------------------------------

**Returns**

True if the position is valid

**3.1.3.5 getBoardSquares()**

```
Square [][] scrabble.Board.getBoardSquares ( )
```

Method to return the current [Board](#)

**Returns**

the current [Board](#)

**3.1.3.6 getSquare()**

```
Square scrabble.Board.getSquare (
    int i,
    int j )
```

Method that returns a specific board square

**Parameters**

<i>i</i>	row coordinate
<i>j</i>	column coordinate

**Returns**

The [Square](#) at i and j

**Exceptions**

<i>InvalidBoardException</i>	Coordinates are not inside the <a href="#">Board</a>
------------------------------	------------------------------------------------------

**3.1.3.7 placeTile()**

```
void scrabble.Board.placeTile (
    Tile tile,
    int position_i,
    int position_j ) [protected]
```

Method to place a [Tile](#) on the board

**Parameters**

<i>tile</i>	<a href="#">Tile</a> to be placed on the <a href="#">Board</a>
<i>position</i> <sub>↔</sub> <i>_i</i>	I position on the <a href="#">Board</a> to place the <a href="#">Tile</a>
<i>position</i> <sub>↔</sub> <i>_j</i>	J position on the <a href="#">Board</a> to place the <a href="#">Tile</a>



### 3.1.3.8 placeTiles()

```
void scrabble.Board.placeTiles (
    MoveInfo moveInfo )
```

Method for a [Player](#) to place a list of Tiles on the [Board](#)

#### Parameters

<i>moveInfo</i>	The move to place on the <a href="#">Board</a>
-----------------	------------------------------------------------

### 3.1.3.9 removeMove()

```
void scrabble.Board.removeMove (
    MoveInfo moveInfo )
```

Method to remove the Tiles place in a move

#### Parameters

<i>moveInfo</i>	The move to remove
-----------------	--------------------

### 3.1.3.10 resetBoard()

```
void scrabble.Board.resetBoard ( )
```

Method to reset the [Board](#)

### 3.1.3.11 setWordSquaresNormal()

```
void scrabble.Board.setWordSquaresNormal (
    Word word )
```

Method to set all Squares under a word to Normal

#### Parameters

<i>word</i>	The <a href="#">Word</a>
-------------	--------------------------

### 3.1.3.12 toString()

```
String scrabble.Board.toString ( )
```

toString method that prints the [Board](#)

#### Returns

Returns the board as a string

### 3.1.4 Member Data Documentation

#### 3.1.4.1 BINGO

```
final int scrabble.Board.BINGO = 50 [static]
```

Bingo Score Bonus

#### 3.1.4.2 BOARD\_SIZE

```
final int scrabble.Board.BOARD_SIZE = 15 [static]
```

Constant value for [Board](#) size

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

- src/main/java/scrabble/Board.java

## 3.2 scrabble.BoardTest Class Reference

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

- src/test/java/scrabble/BoardTest.java

## 3.3 scrabble.UserInput.Direction Enum Reference

### Public Attributes

- VERTICAL
- HORIZONTAL

#### 3.3.1 Detailed Description

direction is an enum type for the types directions a word can be placed on a [Board](#)

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

- src/main/java/scrabble/UserInput.java

## 3.4 scrabble.Frame Class Reference

### Public Member Functions

- [Frame](#) ([Pool](#) pool)
- void [fillFrame](#) ()
- [ArrayList](#)< [Tile](#) > [returnFrame](#) ()
- boolean [isEmpty](#) ()
- void [removeTile](#) (int i)
- void [removeTile](#) (char c)
- void [removeTile](#) ([Tile](#) t)
- void [removeTiles](#) (char[] word)
- void [removeTiles](#) ([ArrayList](#)< [Tile](#) > tiles)
- void [addTile](#) ([Tile](#) tile)
- boolean [checkTiles](#) ([ArrayList](#)< [Tile](#) > tiles)
- boolean [checkTiles](#) (char[] word)
- [Tile](#) [getTile](#) (char c)
- [Tile](#) [getTile](#) (int i)
- [ArrayList](#)< [Tile](#) > [getTiles](#) (char[] word)
- void [swapTiles](#) (char[] tiles)
- void [setBlanks](#) (char[] [tileValues](#))
- void [setToBlank](#) ()
- boolean [hasBlank](#) ()
- int [tileValues](#) ()
- String [toString](#) ()

### Static Public Attributes

- static final int [FRAME\\_SIZE](#) = 7

### 3.4.1 Detailed Description

This class represents the [Player's Frame](#) in [Scrabble](#)

### 3.4.2 Constructor & Destructor Documentation

#### 3.4.2.1 [Frame\(\)](#)

```
scrabble.Frame.Frame (
    Pool pool )
```

Constructor for [Frame](#)

#### Parameters

<i>pool</i>	The reference to the <a href="#">Pool</a> to access Tiles from
-------------	----------------------------------------------------------------

### 3.4.3 Member Function Documentation

#### 3.4.3.1 addTile()

```
void scrabble.Frame.addTile (
    Tile tile )
```

Method to add a single [Tile](#) to the frame

##### Parameters

<i>tile</i>	To be added to the playerFrame
-------------	--------------------------------

##### Exceptions

<i>InvalidFrameException</i>	The <a href="#">Frame</a> is full
------------------------------	-----------------------------------

#### 3.4.3.2 checkTiles() [1/2]

```
boolean scrabble.Frame.checkTiles (
    ArrayList< Tile > tiles )
```

Method which checks if a series of Tiles are currently in the [Frame](#)

##### Parameters

<i>tiles</i>	List of tiles to be checked
--------------	-----------------------------

##### Returns

boolean: Result for if the [Frame](#) contains all the Tiles

##### Exceptions

<i>InvalidFrameException</i>	The <a href="#">Frame</a> is empty
------------------------------	------------------------------------

#### 3.4.3.3 checkTiles() [2/2]

```
boolean scrabble.Frame.checkTiles (
    char[] word )
```

Method to check if a list of characters are in the [Frame](#)

## Parameters

<i>word</i>	List of characters to check
-------------	-----------------------------

## Returns

Boolean answer

## Exceptions

<i>InvalidFrameException</i>	The <a href="#">Frame</a> is empty
------------------------------	------------------------------------

**3.4.3.4 fillFrame()**

```
void scrabble.Frame.fillFrame ( )
```

Method to fill the [Frame](#) up to the maximum number of Tiles

**3.4.3.5 getTile() [1/2]**

```
Tile scrabble.Frame.getTile (
    char c )
```

Method that retrieves a [Tile](#) with a given character from the [Frame](#)

## Parameters

<i>c</i>	Character of <a href="#">Tile</a> wanted
----------	------------------------------------------

## Returns

[Tile](#) with given character

## Exceptions

<i>InvalidFrameException</i>	The <a href="#">Tile</a> is not in the <a href="#">Frame</a>
------------------------------	--------------------------------------------------------------

**3.4.3.6 getTile() [2/2]**

```
Tile scrabble.Frame.getTile (
    int i )
```

Accessor method for [Tile](#) from [Frame](#)

**Parameters**

<i>i</i>	index of the <a href="#">Tile</a> in the <a href="#">Frame</a>
----------	----------------------------------------------------------------

**Returns**

The [Tile](#) at index *i* in the [Frame](#)

**Exceptions**

<i>InvalidFrameException</i>	The
------------------------------	-----

**3.4.3.7 getTiles()**

```
ArrayList<Tile> scrabble.Frame.getTiles (
    char[] word )
```

Method to retrieve a list of [Tile](#) with given characters from [Frame](#)

**Parameters**

<i>word</i>	list of wanted character Tiles
-------------	--------------------------------

**Returns**

List of Tiles

**Exceptions**

<i>InvalidFrameException</i>	The <a href="#">Frame</a> does not contain the Tiles
------------------------------	------------------------------------------------------

**3.4.3.8 hasBlank()**

```
boolean scrabble.Frame.hasBlank ( )
```

Method to check if the [Frame](#) has any blank tiles

**Returns**

Boolean result to if the [Frame](#) has any blank tiles

### 3.4.3.9 isEmpty()

```
boolean scrabble.Frame.isEmpty ( )
```

Method to check if the playerFrame has Tiles in it

#### Returns

boolean: Answer for if the playerFrame is empty

### 3.4.3.10 removeTile() [1/3]

```
void scrabble.Frame.removeTile (
    char c )
```

Method to remove a single [Tile](#) from the [Frame](#)

#### Parameters

<i>c</i>	Character of the <a href="#">Tile</a> to be removed
----------	-----------------------------------------------------

#### Exceptions

<i>InvalidTileException</i>	If Character is not in the <a href="#">Frame</a>
-----------------------------	--------------------------------------------------

### 3.4.3.11 removeTile() [2/3]

```
void scrabble.Frame.removeTile (
    int i )
```

Method to remove a single [Tile](#) from the [Frame](#)

#### Parameters

<i>i</i>	<a href="#">Tile</a> index to be removed
----------	------------------------------------------

#### Exceptions

<i>InvalidTileException</i>	If index is not in the <a href="#">Frame</a>
-----------------------------	----------------------------------------------

#### 3.4.3.12 removeTile() [3/3]

```
void scrabble.Frame.removeTile (
    Tile t )
```

Method to remove a single [Tile](#) from the [Frame](#)

##### Parameters

<i>t</i>	<a href="#">Tile</a> to be removed
----------	------------------------------------

#### 3.4.3.13 removeTiles() [1/2]

```
void scrabble.Frame.removeTiles (
    ArrayList< Tile > tiles )
```

Method to remove a list of Tiles from the [Frame](#)

##### Parameters

<i>tiles</i>	List of Tiles to be removed
--------------	-----------------------------

##### Exceptions

<i>InvalidFrameException</i>	If the Tiles to be removed are in the <a href="#">Frame</a>
------------------------------	-------------------------------------------------------------

#### 3.4.3.14 removeTiles() [2/2]

```
void scrabble.Frame.removeTiles (
    char[] word )
```

Method to remove a multiple [Tile](#) from the [Frame](#)

##### Parameters

<i>word</i>	Array of <a href="#">Tile</a> Characters to be removed
-------------	--------------------------------------------------------

##### Exceptions

<i>InvalidTileException</i>	If <a href="#">Tile</a> is not in the <a href="#">Frame</a>
-----------------------------	-------------------------------------------------------------



#### 3.4.3.15 returnFrame()

```
ArrayList<Tile> scrabble.Frame.returnFrame ( )
```

Accessor method for playerFrame

##### Returns

playerFrame: The ArrayList of Tiles in playerFrame

#### 3.4.3.16 setBlanks()

```
void scrabble.Frame.setBlanks (
    char[] tileValues )
```

Method to set the Blank Tiles in [Frame](#)

##### Parameters

<i>tileValues</i>	The chars to change the Blank Tiles to
-------------------	----------------------------------------

##### Exceptions

<i>InvalidFrameException</i>	If too many chars are passed
------------------------------	------------------------------

#### 3.4.3.17 setToBlank()

```
void scrabble.Frame.setToBlank ( )
```

Method to set all Blank Tiles to Null

#### 3.4.3.18 swapTiles()

```
void scrabble.Frame.swapTiles (
    char[] tiles )
```

Method used to swap a number of Tiles from the [Frame](#) for new ones in the [Pool](#)

##### Parameters

<i>tiles</i>	List of Tiles to be removed
--------------	-----------------------------

#### 3.4.3.19 tileValues()

```
int scrabble.Frame.tileValues ( )
```

Method to find the total value of the Tiles in a [Frame](#)

##### Returns

The total value of the Tiles

#### 3.4.3.20 toString()

```
String scrabble.Frame.toString ( )
```

Method overriding the toString method

##### Returns

String: Formatted string of the Tiles in [Frame](#)

### 3.4.4 Member Data Documentation

#### 3.4.4.1 FRAME\_SIZE

```
final int scrabble.Frame.FRAME_SIZE = 7 [static]
```

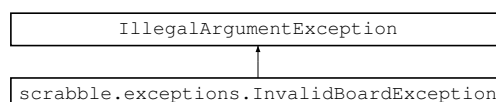
Max amount of Tiles in the [Frame](#)

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

- src/main/java/scrabble/Frame.java

## 3.5 scrabble.exceptions.InvalidBoardException Class Reference

Inheritance diagram for scrabble.exceptions.InvalidBoardException:



## Public Member Functions

- **InvalidBoardException** (String s)

### 3.5.1 Detailed Description

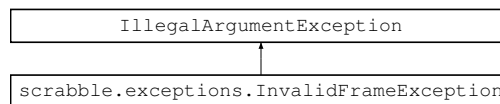
Custom Exception for [Board](#) Class

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

- src/main/java/scrabble/exceptions/InvalidBoardException.java

## 3.6 scrabble.exceptions.InvalidFrameException Class Reference

Inheritance diagram for scrabble.exceptions.InvalidFrameException:



## Public Member Functions

- **InvalidFrameException** (String s)

### 3.6.1 Detailed Description

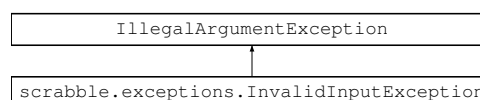
Custom Exception for [Frame](#) Class

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

- src/main/java/scrabble/exceptions/InvalidFrameException.java

## 3.7 scrabble.exceptions.InvalidInputException Class Reference

Inheritance diagram for scrabble.exceptions.InvalidInputException:



## Public Member Functions

- **InvalidInputException** (String s)

### 3.7.1 Detailed Description

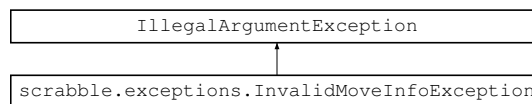
Custom exception for [UserInput](#) Class

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

- src/main/java/scrabble/exceptions/InvalidInputException.java

## 3.8 `scrabble.exceptions.InvalidMoveInfoException` Class Reference

Inheritance diagram for `scrabble.exceptions.InvalidMoveInfoException`:



## Public Member Functions

- **InvalidMoveInfoException** (String s)

### 3.8.1 Detailed Description

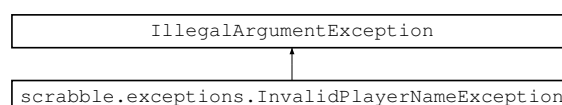
Custom Exception for [MoveInfo](#) Class

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

- src/main/java/scrabble/exceptions/InvalidMoveInfoException.java

## 3.9 `scrabble.exceptions.InvalidPlayerNameException` Class Reference

Inheritance diagram for `scrabble.exceptions.InvalidPlayerNameException`:



## Public Member Functions

- **InvalidPlayerNameException** (String s)

### 3.9.1 Detailed Description

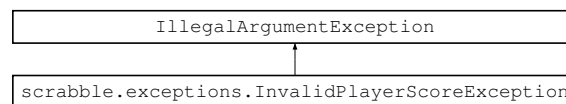
Custom Exception for Name in [Player](#) Class

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

- src/main/java/scrabble/exceptions/InvalidPlayerNameException.java

## 3.10 scrabble.exceptions.InvalidPlayerScoreException Class Reference

Inheritance diagram for scrabble.exceptions.InvalidPlayerScoreException:



## Public Member Functions

- **InvalidPlayerScoreException** (String s)

### 3.10.1 Detailed Description

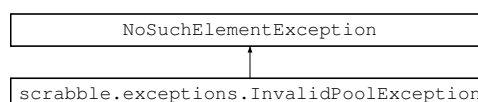
Custom Exception for Score in [Player](#) Class

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

- src/main/java/scrabble/exceptions/InvalidPlayerScoreException.java

## 3.11 scrabble.exceptions.InvalidPoolException Class Reference

Inheritance diagram for scrabble.exceptions.InvalidPoolException:



## Public Member Functions

- **InvalidPoolException** (String s)

### 3.11.1 Detailed Description

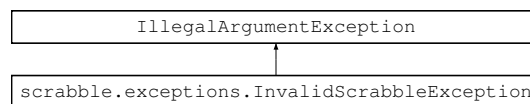
Custom Exception for [Pool](#) Class

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

- src/main/java/scrabble/exceptions/InvalidPoolException.java

## 3.12 `scrabble.exceptions.InvalidScrabbleException` Class Reference

Inheritance diagram for `scrabble.exceptions.InvalidScrabbleException`:



## Public Member Functions

- **InvalidScrabbleException** (String s)

### 3.12.1 Detailed Description

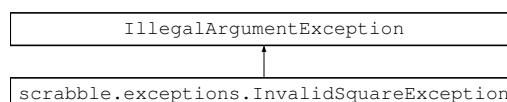
Custom Exception for [Scrabble](#) Class

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

- src/main/java/scrabble/exceptions/InvalidScrabbleException.java

## 3.13 `scrabble.exceptions.InvalidSquareException` Class Reference

Inheritance diagram for `scrabble.exceptions.InvalidSquareException`:



## Public Member Functions

- **InvalidSquareException** (String s)

### 3.13.1 Detailed Description

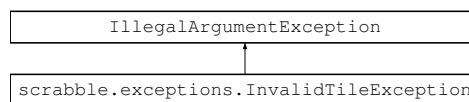
Custom Exception for [Square](#) Class

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

- src/main/java/scrabble/exceptions/InvalidSquareException.java

## 3.14 scrabble.exceptions.InvalidTileException Class Reference

Inheritance diagram for scrabble.exceptions.InvalidTileException:



## Public Member Functions

- **InvalidTileException** (String s)

### 3.14.1 Detailed Description

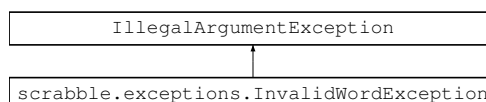
Custom Exception for [Tile](#) Class

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

- src/main/java/scrabble/exceptions/InvalidTileException.java

## 3.15 scrabble.exceptions.InvalidWordException Class Reference

Inheritance diagram for scrabble.exceptions.InvalidWordException:



## Public Member Functions

- **InvalidWordException** (String s)

### 3.15.1 Detailed Description

Custom Exception for [Word](#) Class

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

- src/main/java/scrabble/exceptions/InvalidWordException.java

## 3.16 scrabble.Main Class Reference

### Static Public Member Functions

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

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

- src/main/java/scrabble/Main.java

## 3.17 scrabble.MoveInfo Class Reference

### Public Member Functions

- ArrayList< [Word](#) > [getAuxiliaryWords](#) ()
- char[] [getRequiredTiles](#) ()
- int[][] [getRequiredTilesPositions](#) ()
- int [getMoveScore](#) ()
- [Word](#) [getPrimaryWord](#) ()
- [Player](#) [getPlayer](#) ()
- [MoveInfo](#) ([Player](#) p, int[] coOrdinates, [UserInput.Direction](#) d, char[] w)
- void [addAuxiliaryWord](#) ([Word](#) auxWord)
- void [setRequiredTiles](#) (char[] requiredTiles, int[][] requiredTilesPositions)

### Protected Member Functions

- void [setScore](#) (int score)

### 3.17.1 Detailed Description

Object to store data on a move



## 3.17.2 Constructor & Destructor Documentation

### 3.17.2.1 MoveInfo()

```
scrabble.MoveInfo.MoveInfo (
    Player p,
    int[] coOrdinates,
    UserInput.Direction d,
    char[] w )
```

[MoveInfo](#) Constructor

## Parameters

<i>p</i>	<a href="#">Player</a> making the move
<i>coOrdinates</i>	Position of the first letter of the <a href="#">Word</a>
<i>d</i>	Direction of the <a href="#">Word</a>
<i>w</i>	<a href="#">Word</a> in char array form

## Exceptions

<i>InvalidMoveInfoException</i>	<a href="#">Player</a> can not be Null
---------------------------------	----------------------------------------

### 3.17.3 Member Function Documentation

#### 3.17.3.1 addAuxiliaryWord()

```
void scrabble.MoveInfo.addAuxiliaryWord (
    Word auxWord )
```

Method to add auxiliary Words to the move

## Parameters

<i>auxWord</i>	<a href="#">Word</a> to be added
----------------	----------------------------------

#### 3.17.3.2 getAuxiliaryWords()

```
ArrayList<Word> scrabble.MoveInfo.getAuxiliaryWords ( )
```

Accessor Method for auxiliaryWords

## Returns

auxiliaryWords

#### 3.17.3.3 getMoveScore()

```
int scrabble.MoveInfo.getMoveScore ( )
```

Accessor Method for moveScore

## Returns

moveScore

#### 3.17.3.4 getPlayer()

```
Player scrabble.MoveInfo.getPlayer ( )
```

Accessor Method for [Player](#)

Returns

[Player](#)

#### 3.17.3.5 getPrimaryWord()

```
Word scrabble.MoveInfo.getPrimaryWord ( )
```

Accessor Method for moveScore

Returns

moveScore

#### 3.17.3.6 getRequiredTiles()

```
char [ ] scrabble.MoveInfo.getRequiredTiles ( )
```

Accessor Method for requiredTiles

Returns

requiredTiles

#### 3.17.3.7 getRequiredTilesPositions()

```
int [][ ] scrabble.MoveInfo.getRequiredTilesPositions ( )
```

Accessor Method for requiredTilesPositions

Returns

requiredTilesPositions

#### 3.17.3.8 setRequiredTiles()

```
void scrabble.MoveInfo.setRequiredTiles (
    char[] requiredTiles,
    int requiredTilesPositions[][ ] )
```

Mutator Method for requiredTiles and requiredTilesPositions

## Parameters

<i>requiredTiles</i>	The Tiles required for the move
<i>requiredTilesPositions</i>	The positions for the Tiles on the <a href="#">Board</a>

**3.17.3.9 setScore()**

```
void scrabble.MoveInfo.setScore (
    int score ) [protected]
```

Mutator Method for Score

## Parameters

<i>score</i>	The Score of the move
--------------	-----------------------

## Exceptions

<i>InvalidMoveInfoException</i>	The move must have a positive score
---------------------------------	-------------------------------------

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

- `src/main/java/scrabble/MoveInfo.java`

**3.18 scrabble.MoveInfoTest Class Reference**

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

- `src/test/java/scrabble/MoveInfoTest.java`

**3.19 scrabble.Player Class Reference****Public Member Functions**

- [Player](#) (String namePlayer, [Pool](#) pool)
- void [playerReset](#) (String newName)
- String [getName](#) ()
- void [setName](#) (String name)
- int [getScore](#) ()
- void [increaseScore](#) (int scoreIncrease)
- void [decreaseScore](#) (int scoreDecrease)
- [Frame](#) [getPlayerFrame](#) ()
- boolean [charUserInputCheck](#) (char letter)
- String [toString](#) ()

### 3.19.1 Detailed Description

Class that represents a player. Contains the information of each players

### 3.19.2 Constructor & Destructor Documentation

#### 3.19.2.1 Player()

```
scrabble.Player.Player (
    String namePlayer,
    Pool pool )
```

[Player](#) Constructor

##### Parameters

<i>namePlayer</i>	String for the name of the <a href="#">Player</a>
<i>pool</i>	Reference to the <a href="#">Pool</a> of the game

##### Exceptions

<i>InvalidPlayerNameException</i>	If inputted name is invalid
-----------------------------------	-----------------------------

### 3.19.3 Member Function Documentation

#### 3.19.3.1 decreaseScore()

```
void scrabble.Player.decreaseScore (
    int scoreDecrease )
```

Mutator method for score to decrease the players score

##### Parameters

<i>scoreDecrease</i>	The value for the score to be increased by
----------------------	--------------------------------------------

##### Exceptions

<i>InvalidPlayerScoreException</i>	If a negative value is passed into Decrease_Score
------------------------------------	---------------------------------------------------

### 3.19.3.2 getName()

```
String scrabble.Player.getName ( )
```

Accessor Method for Name

#### Returns

The name of the player

### 3.19.3.3 getPlayerFrame()

```
Frame scrabble.Player.getPlayerFrame ( )
```

Accessing Method player's frame

#### Returns

The value of playerFrame

### 3.19.3.4 getScore()

```
int scrabble.Player.getScore ( )
```

Accessor Method for [Player](#) Score

#### Returns

The [Player](#)'s Score

### 3.19.3.5 increaseScore()

```
void scrabble.Player.increaseScore (
    int scoreIncrease )
```

Mutator method for score to increase the players score

#### Parameters

<i>scoreIncrease</i>	The value for the score to be increased by
----------------------	--------------------------------------------

## Exceptions

<i>InvalidPlayerScoreException</i>	If player's score is increased by a negative value
------------------------------------	----------------------------------------------------

**3.19.3.6 playerReset()**

```
void scrabble.Player.playerReset (
    String newName )
```

Resetting the score and name of the player

## Parameters

<i>newName</i>	The new player name after the player is reset
----------------	-----------------------------------------------

## Exceptions

<i>InvalidPlayerNameException</i>	If inputted name is invalid
-----------------------------------	-----------------------------

**3.19.3.7 setName()**

```
void scrabble.Player.setName (
    String name )
```

Mutator Method for name

## Parameters

<i>name</i>	The new Name of the <a href="#">Player</a>
-------------	--------------------------------------------

## Exceptions

<i>InvalidPlayerNameException</i>	If inputted name is invalid
-----------------------------------	-----------------------------

**3.19.3.8 toString()**

```
String scrabble.Player.toString ( )
```

A toString method to print the [Player](#) class variables

**Returns**

The [Player](#) class variables

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

- `src/main/java/scrabble/Player.java`

## 3.20 scrabble.Pool Class Reference

### Public Member Functions

- void [poolFill](#) () throws InvalidTileException
- String [toString](#) ()
- [Tile](#) [removeTile](#) () throws InvalidPoolException
- void [receiveTile](#) ([Tile](#) tileAdded)
- int [tilesInPool](#) ()
- boolean [isEmpty](#) ()
- [Pool](#) ()

### Static Public Member Functions

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

#### 3.20.1 Detailed Description

Class that represent a Character [Tile](#) for the Game of [Scrabble](#)

#### 3.20.2 Constructor & Destructor Documentation

##### 3.20.2.1 [Pool](#)()

```
scrabble.Pool.Pool ( )
```

[Pool](#) constructor

#### 3.20.3 Member Function Documentation



### 3.20.3.1 isEmpty()

```
boolean scrabble.Pool.isEmpty ( )
```

Method to Check if the pool is empty

#### Returns

True if pool is empty

### 3.20.3.2 poolFill()

```
void scrabble.Pool.poolFill ( ) throws InvalidTileException
```

Function to fill the array with the set amount of each [Tile](#) in the standard English rules

### 3.20.3.3 receiveTile()

```
void scrabble.Pool.receiveTile (
    Tile tileAdded )
```

Method to take in a tile and add it to a pool

#### Parameters

<i>tileAdded</i>	The tile to be added to the pool
------------------	----------------------------------

### 3.20.3.4 removeTile()

```
Tile scrabble.Pool.removeTile ( ) throws InvalidPoolException
```

Method to remove a random tile from the pool and return the tile that was removed

#### Returns

The tile which was randomly removed from the pool

#### Exceptions

<i>InvalidPoolException</i>	If the pool is empty
-----------------------------	----------------------

### 3.20.3.5 tilesInPool()

```
int scrabble.Pool.tilesInPool ( )
```

Method to find the number of tiles in pool

#### Returns

The number of tiles in pool

### 3.20.3.6 toString()

```
String scrabble.Pool.toString ( )
```

[Pool](#) toString Method

#### Returns

[Pool](#) information in the form of a String

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

- src/main/java/scrabble/Pool.java

## 3.21 scrabble.PoolTest Class Reference

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

- src/test/java/scrabble/PoolTest.java

## 3.22 scrabble.Scrabble Class Reference

### Public Member Functions

- [Scrabble](#) () throws FileNotFoundException
- [Board](#) [getBoard](#) ()
- [Pool](#) [getPool](#) ()
- [Player](#)[] [getPlayers](#) ()
- ArrayList< [MoveInfo](#) > [getMoveHistory](#) ()
- void [createPlayer](#) (String name, int playerNumber)
- void [playerMove](#) (int[] startPosition, [UserInput.Direction](#) direction, char[] word, [Player](#) player)
- boolean [isGameOver](#) ()
- void [gameOver](#) ()
- boolean [dictionaryWords](#) ([MoveInfo](#) move)

### 3.22.1 Constructor & Destructor Documentation

#### 3.22.1.1 Scrabble()

```
scrabble.Scrabble.Scrabble ( ) throws FileNotFoundException
```

[Scrabble](#) Game Constructor

Creates a new game of [Scrabble](#)

## Exceptions

<i>FileNotFoundException</i>	If Scanner fails to find <a href="#">Scrabble</a> Dictionary
------------------------------	--------------------------------------------------------------

## 3.22.2 Member Function Documentation

### 3.22.2.1 createPlayer()

```
void scrabble.Scrabble.createPlayer (
    String name,
    int playerNumber )
```

Method to create a player with an inputted name

## Parameters

<i>name</i>	The name of the player
<i>playerNumber</i>	The index of the player in the players array

## Exceptions

<i>InvalidScrabbleException</i>	If the index is out of bounds of the players array
---------------------------------	----------------------------------------------------

### 3.22.2.2 dictionaryWords()

```
boolean scrabble.Scrabble.dictionaryWords (
    MoveInfo move )
```

Method to validate the words created in a move

## Parameters

<i>move</i>	The move to validate
-------------	----------------------

## Returns

True if all words are valid

### 3.22.2.3 gameOver()

```
void scrabble.Scrabble.gameOver ( )
```

Method to subtract [Frame Tile](#) values from Players' scores at the end of the game

### 3.22.2.4 getBoard()

```
Board scrabble.Scrabble.getBoard ( )
```

Accessor Method for [Board](#)

#### Returns

The [Board](#)

### 3.22.2.5 getMoveHistory()

```
ArrayList<MoveInfo> scrabble.Scrabble.getMoveHistory ( )
```

Accessor Method for MoveHistory

#### Returns

The MoveHistory ArrayList

### 3.22.2.6 getPlayers()

```
Player [ ] scrabble.Scrabble.getPlayers ( )
```

Accessor Method for Players

#### Returns

The Players array

### 3.22.2.7 getPool()

```
Pool scrabble.Scrabble.getPool ( )
```

Accessor Method for [Pool](#)

#### Returns

The Poll

### 3.22.2.8 isGameOver()

```
boolean scrabble.Scrabble.isGameOver ( )
```

Method to check if the game is over

#### Returns

True if the game is over

### 3.22.2.9 playerMove()

```
void scrabble.Scrabble.playerMove (
    int[] startPosition,
    UserInput.Direction direction,
    char[] word,
    Player player )
```

Method to complete a [Player](#) move

#### Parameters

<i>startPosition</i>	Start position of the <a href="#">Word</a>
<i>direction</i>	Direction of the <a href="#">Word</a>
<i>word</i>	<a href="#">Word</a> in char array form
<i>player</i>	<a href="#">Player</a> making the move

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

- `src/main/java/scrabble/Scrabble.java`

## 3.23 scrabble.ScrabbleTest Class Reference

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

- `src/test/java/scrabble/ScrabbleTest.java`

## 3.24 scrabble.Square Class Reference

### Classes

- enum [SquareType](#)

## Public Member Functions

- [Square](#) ([SquareType](#) type)
- [SquareType](#) [getType](#) ()
- [Tile](#) [getTile](#) ()
- Boolean [isEmpty](#) ()
- [SquareType](#) [setNormal](#) ()
- void [setTile](#) ([Tile](#) tile)
- String [toString](#) ()
- [Tile](#) [setEmpty](#) ()

### 3.24.1 Detailed Description

The [Square](#) Class represents the square on the [Scrabble Board](#) as objects

### 3.24.2 Constructor & Destructor Documentation

#### 3.24.2.1 [Square](#)()

```
scrabble.Square.Square (  
    SquareType type )
```

[Square](#) Constructor

Parameters

<i>type</i>	The <a href="#">SquareType</a> of the <a href="#">Square</a>
-------------	--------------------------------------------------------------

### 3.24.3 Member Function Documentation

#### 3.24.3.1 [getTile](#)()

```
Tile scrabble.Square.getTile ( )
```

Accessor Method for the [Tile](#) on the [Square](#)

Returns

The [Tile](#) on the [Square](#)

### 3.24.3.2 getType()

```
SquareType scrabble.Square.getType ( )
```

Accessor Method for the [SquareType](#) of the [Square](#)

#### Returns

The [SquareType](#) of the [Square](#)

### 3.24.3.3 isEmpty()

```
Boolean scrabble.Square.isEmpty ( )
```

Method to find if the [Square](#) has a [Tile](#)

#### Returns

True if the [Square](#) has no [Tile](#) on it

### 3.24.3.4 setEmpty()

```
Tile scrabble.Square.setEmpty ( )
```

Method to setSquare to empty

#### Returns

The [Tile](#) on the [Square](#)

#### Exceptions

<i>InvalidSquareException</i>	The <a href="#">Square</a> was empty
-------------------------------	--------------------------------------

### 3.24.3.5 setNormal()

```
SquareType scrabble.Square.setNormal ( )
```

Method to set the [SquareType](#) of a [Square](#) to NORMAL

#### Returns

The [SquareType](#) of the [Square](#) before setting to normal

### 3.24.3.6 setTile()

```
void scrabble.Square.setTile (
    Tile tile )
```

Mutator Method for [Tile](#) on [Square](#)

#### Parameters

<i>tile</i>	<a href="#">Tile</a> to be placed on the <a href="#">Square</a>
-------------	-----------------------------------------------------------------

#### Exceptions

<i>InvalidSquareException</i>	The <a href="#">Square</a> has a <a href="#">Tile</a> on it already or the <a href="#">Tile</a> is a null <a href="#">Tile</a>
-------------------------------	--------------------------------------------------------------------------------------------------------------------------------

### 3.24.3.7 toString()

```
String scrabble.Square.toString ( )
```

toString Method for [Square](#)

#### Returns

the String form of [Square](#)

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

- `src/main/java/scrabble/Square.java`

## 3.25 scrabble.SquareTest Class Reference

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

- `src/test/java/scrabble/SquareTest.java`

## 3.26 scrabble.Square.SquareType Enum Reference

### Public Attributes

- `NORMAL`
- `START`
- `DOUBLE_WORD`
- `TRIPLE_WORD`
- `DOUBLE_LETTER`
- `TRIPLE_LETTER`



### 3.26.1 Detailed Description

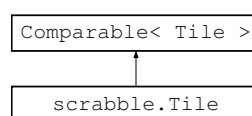
[SquareType](#) is an Enum of possible Types of [Square](#) on the [Scrabble Board](#)

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

- `src/main/java/scrabble/Square.java`

## 3.27 scrabble.Tile Class Reference

Inheritance diagram for `scrabble.Tile`:



### Public Member Functions

- `int` [compareTo](#) ([Tile](#) t)
- `boolean` [equals](#) (Object obj)
- `short` [getValue](#) ()
- `char` [getCharacter](#) ()
- `void` [setCharacter](#) (char character) throws `InvalidTileException`
- `void` [setNull](#) ()
- [Tile](#) (char c) throws `InvalidTileException`
- `String` [toString](#) ()

### Static Public Member Functions

- `static void` [main](#) (String[] args)

### 3.27.1 Detailed Description

Class that represent a Character [Tile](#) for the Game of [Scrabble](#)

### 3.27.2 Constructor & Destructor Documentation

#### 3.27.2.1 [Tile](#)()

```
scrabble.Tile.Tile (
    char c ) throws InvalidTileException
```

Constructor for [Tile](#)

**Parameters**

<i>c</i>	Character for the <a href="#">Tile</a>
----------	----------------------------------------

**Exceptions**

<i>InvalidTileException</i>	If an invalid character is inputted
-----------------------------	-------------------------------------

### 3.27.3 Member Function Documentation

#### 3.27.3.1 compareTo()

```
int scrabble.Tile.compareTo (
    Tile t )
```

Method to compare [Tile](#) objects by Character then Value

**Parameters**

<i>t</i>	<a href="#">Tile</a> for this tile to be compared to
----------	------------------------------------------------------

**Returns**

Returns 0 if equal. Returns 1 if this Character is greater or Characters are equals and this Value is greater else returns -1

#### 3.27.3.2 equals()

```
boolean scrabble.Tile.equals (
    Object obj )
```

Method to see if this [Tile](#) equals another object

**Parameters**

<i>obj</i>	Object to be compared to
------------	--------------------------

**Returns**

Returns True if the objects are equal

### 3.27.3.3 getCharacter()

```
char scrabble.Tile.getCharacter ( )
```

Accessor method for character of the [Tile](#)

#### Returns

Char of the [Tile](#)

### 3.27.3.4 getValue()

```
short scrabble.Tile.getValue ( )
```

Accessor method for value of the tile

#### Returns

Short value of the tile

### 3.27.3.5 main()

```
static void scrabble.Tile.main (
    String[] args ) [static]
```

[Main](#) Function

#### Parameters

<i>args</i>	Arguments
-------------	-----------

### 3.27.3.6 setCharacter()

```
void scrabble.Tile.setCharacter (
    char character ) throws InvalidTileException
```

Method to change the character of a blank [Tile](#)

#### Parameters

<i>character</i>	New char for the <a href="#">Tile</a>
------------------	---------------------------------------

**Exceptions**

<i>InvalidTileException</i>	If an invalid char inputted or the <a href="#">Tile</a> is not blank
-----------------------------	----------------------------------------------------------------------

**3.27.3.7 setNull()**

```
void scrabble.Tile.setNull ( )
```

Method to set Blank Tiles Character back to null

**Exceptions**

<i>InvalidTileException</i>	If the tile is not a blank tile
-----------------------------	---------------------------------

**3.27.3.8 toString()**

```
String scrabble.Tile.toString ( )
```

toString Method for [Tile](#)

**Returns**

The string of [Tile](#)

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

- src/main/java/scrabble/Tile.java

**3.28 scrabble.UserInput Class Reference****Classes**

- enum [Direction](#)
- enum [UserInputType](#)

**Public Member Functions**

- [UserInput](#) ([UserInputType](#) type)
- [UserInput](#) ([UserInputType](#) type, char[] tileExchange)
- [UserInput](#) ([UserInputType](#) type, String UserName)
- [UserInput](#) ([UserInputType](#) type, char[] w, int[] position, [Direction](#) d)
- [UserInputType](#) [getInputType](#) ()
- char[] [getWord](#) ()
- String [getName](#) ()
- int[] [getStartPosition](#) ()
- [Direction](#) [getWordDirection](#) ()

## Static Public Member Functions

- static [UserInput parseInput](#) (String input)

### 3.28.1 Constructor & Destructor Documentation

#### 3.28.1.1 UserInput() [1/4]

```
scrabble.UserInput.UserInput (
    UserInputType type )
```

Constructor for cases with one token such as 'HELP' or 'PASS'

##### Parameters

<i>type</i>	The <a href="#">UserInputType</a>
-------------	-----------------------------------

#### 3.28.1.2 UserInput() [2/4]

```
scrabble.UserInput.UserInput (
    UserInputType type,
    char[] tileExchange )
```

Constructor for when user is swapping a Tile(s)

##### Parameters

<i>type</i>	The <a href="#">UserInputType</a>
<i>tileExchange</i>	An array of characters that are to be swapped

##### Exceptions

<i>InvalidInputException</i>	If user input are not valid tiles
------------------------------	-----------------------------------

#### 3.28.1.3 UserInput() [3/4]

```
scrabble.UserInput.UserInput (
    UserInputType type,
    String UserName )
```

## Parameters

<i>type</i>	The <a href="#">UserInputType</a>
<i>UserName</i>	A String that the User inputted for their name

**3.28.1.4 UserInput()** [4/4]

```
scrabble.UserInput.UserInput (
    UserInputType type,
    char[] w,
    int[] position,
    Direction d )
```

Constructor for when the user wants to place a word on the [Board](#)

## Parameters

<i>type</i>	The <a href="#">UserInputType</a>
<i>w</i>	The word that the user wants to place on the <a href="#">Board</a>
<i>position</i>	The position of the coordinate of the first <a href="#">Tile</a> of the word
<i>d</i>	The direction of the word

**3.28.2 Member Function Documentation****3.28.2.1 getInputType()**

```
UserInputType scrabble.UserInput.getInputType ( )
```

Accessor method for getting the type of input

## Returns

Returns the input type

**3.28.2.2 getName()**

```
String scrabble.UserInput.getName ( )
```

Accessor method for getting the name of the player that the user inputted

## Returns

Returns a string which contains the inputted user name

### 3.28.2.3 `getStartPosition()`

```
int [] scrabble.UserInput.getStartPosition ( )
```

Accessor method for getting the coordinates of the first [Tile](#) of the word the user wants to place on the board

#### Returns

Returns the coordinates of the first [Tile](#) of the word the user wants to place on the board

### 3.28.2.4 `getWord()`

```
char [] scrabble.UserInput.getWord ( )
```

Accessor method for getting the word when placing a [Tile](#). Also used for accessing the Tiles a user wants to swap.

#### Returns

Returns an array of characters which contains either the word the user wants to place or the Tiles the user wants to swap

### 3.28.2.5 `getWordDirection()`

```
Direction scrabble.UserInput.getWordDirection ( )
```

Accessor method for getting the direction of the word that is to be placed on the [Board](#)

#### Returns

Returns the direction of the word that is to be placed on the [Board](#)

### 3.28.2.6 `parseInput()`

```
static UserInput scrabble.UserInput.parseInput (
    String input ) [static]
```

#### Parameters

<i>input</i>	The string that the User inputted into the FX console which will be broken down and parsed
--------------	--------------------------------------------------------------------------------------------

#### Returns

Returns an object [UserInput](#) which contains the user's input which has been broken down and parsed

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

- `src/main/java/scrabble/UserInput.java`

### 3.29 `scrabble.UserInputTest` Class Reference

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

- `src/test/java/scrabble/UserInputTest.java`

### 3.30 `scrabble.UserInput.UserInputType` Enum Reference

#### Public Attributes

- `QUIT`
- `HELP`
- `PASS`
- `EXCHANGE`
- `PLACE_TILE`
- `ERROR`
- `BLANK`
- `CHALLENGE`
- `RESTART`
- `NAME`

#### 3.30.1 Detailed Description

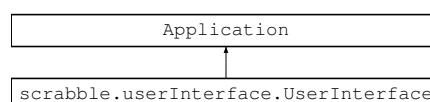
[UserInputType](#) is an enum type for the types of possible expected user inputs

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

- `src/main/java/scrabble/UserInput.java`

### 3.31 `scrabble.userInterface.UserInterface` Class Reference

Inheritance diagram for `scrabble.userInterface.UserInterface`:





## Public Member Functions

- void [start](#) (Stage gameStage)

## Static Public Member Functions

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

### 3.31.1 Member Function Documentation

#### 3.31.1.1 [main\(\)](#)

```
static void scrabble.userInterface.UserInterface.main (  
    String[] args ) [static]
```

[Main](#) method to launch application

Parameters

<i>args</i>	empty argument
-------------	----------------

#### 3.31.1.2 [start\(\)](#)

```
void scrabble.userInterface.UserInterface.start (  
    Stage gameStage )
```

Method to start the javaFx application

Parameters

<i>gameStage</i>	Stage to be used in the application
------------------	-------------------------------------

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

- src/main/java/scrabble/userInterface/UserInterface.java

## 3.32 scrabble.Word Class Reference

### Public Member Functions

- [Word](#) (int[] coOrdinates, UserInput.Direction d, char[] w)

- char[] [getWord](#) ()
- UserInput.Direction [getDirection](#) ()
- int[] [getStartPosition](#) ()
- String [toString](#) ()

### 3.32.1 Detailed Description

[Word](#) Class to store information about a word on the board

### 3.32.2 Constructor & Destructor Documentation

#### 3.32.2.1 Word()

```
scrabble.Word.Word (
    int[] coOrdinates,
    UserInput.Direction d,
    char[] w )
```

[Word](#) Constructor

Parameters

<i>coOrdinates</i>	The starting position on the board (row, col)
<i>d</i>	The direction of the <a href="#">Word</a>
<i>w</i>	The word in char array form

Exceptions

<i>InvalidWordException</i>	The <a href="#">Word</a> must be between 1 - 15 letters
-----------------------------	---------------------------------------------------------

### 3.32.3 Member Function Documentation

#### 3.32.3.1 getDirection()

```
UserInput.Direction scrabble.Word.getDirection ( )
```

Accessor Method for the Direction

Returns

The Direction of the [Word](#)

### 3.32.3.2 `getStartPosition()`

```
int [] scrabble.Word.getStartPosition ( )
```

Accessor Method for the Start Position

#### Returns

The co-ordinates for the Start Position (row, col)

### 3.32.3.3 `getWord()`

```
char [] scrabble.Word.getWord ( )
```

Accessor Method for the [Word](#)

#### Returns

The word in char array form

### 3.32.3.4 `toString()`

```
String scrabble.Word.toString ( )
```

`toString` method for [Word](#)

#### Returns

[Word](#) in string form

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

- `src/main/java/scrabble/Word.java`

## 3.33 scrabble.WordTest Class Reference

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

- `src/test/java/scrabble/WordTest.java`



# Index

- addAuxiliaryWord
  - scrabble.MoveInfo, [26](#)
- addTile
  - scrabble.Frame, [12](#)
- BINGO
  - scrabble.Board, [10](#)
- Board
  - scrabble.Board, [6](#)
- BOARD\_SIZE
  - scrabble.Board, [10](#)
- calculateScore
  - scrabble.Board, [6](#)
- checkPlayerHasTiles
  - scrabble.Board, [6](#)
- checkTiles
  - scrabble.Frame, [12](#)
- checkValidMove
  - scrabble.Board, [7](#)
- checkValidPosition
  - scrabble.Board, [7](#)
- compareTo
  - scrabble.Tile, [42](#)
- createPlayer
  - scrabble.Scrabble, [35](#)
- decreaseScore
  - scrabble.Player, [29](#)
- dictionaryWords
  - scrabble.Scrabble, [35](#)
- equals
  - scrabble.Tile, [42](#)
- fillFrame
  - scrabble.Frame, [13](#)
- Frame
  - scrabble.Frame, [11](#)
- FRAME\_SIZE
  - scrabble.Frame, [18](#)
- gameOver
  - scrabble.Scrabble, [35](#)
- getAuxiliaryWords
  - scrabble.MoveInfo, [26](#)
- getBoard
  - scrabble.Scrabble, [36](#)
- getBoardSquares
  - scrabble.Board, [7](#)
- getCharacter
  - scrabble.Tile, [42](#)
- getDirection
  - scrabble.Word, [50](#)
- getInputType
  - scrabble.UserInput, [46](#)
- getMoveHistory
  - scrabble.Scrabble, [36](#)
- getMoveScore
  - scrabble.MoveInfo, [26](#)
- getName
  - scrabble.Player, [30](#)
  - scrabble.UserInput, [46](#)
- getPlayer
  - scrabble.MoveInfo, [26](#)
- getPlayerFrame
  - scrabble.Player, [30](#)
- getPlayers
  - scrabble.Scrabble, [36](#)
- getPool
  - scrabble.Scrabble, [36](#)
- getPrimaryWord
  - scrabble.MoveInfo, [27](#)
- getRequiredTiles
  - scrabble.MoveInfo, [27](#)
- getRequiredTilesPositions
  - scrabble.MoveInfo, [27](#)
- getScore
  - scrabble.Player, [30](#)
- getSquare
  - scrabble.Board, [8](#)
- getStartPosition
  - scrabble.UserInput, [46](#)
  - scrabble.Word, [50](#)
- getTile
  - scrabble.Frame, [13](#)
  - scrabble.Square, [38](#)
- getTiles
  - scrabble.Frame, [14](#)
- getType
  - scrabble.Square, [38](#)
- getValue
  - scrabble.Tile, [43](#)
- getWord
  - scrabble.UserInput, [47](#)
  - scrabble.Word, [51](#)
- getWordDirection
  - scrabble.UserInput, [47](#)
- hasBlank
  - scrabble.Frame, [14](#)

- increaseScore
  - scrabble.Player, 30
- isEmpty
  - scrabble.Frame, 14
  - scrabble.Pool, 32
  - scrabble.Square, 39
- isGameOver
  - scrabble.Scrabble, 36
- main
  - scrabble.Tile, 43
  - scrabble.userInterface.UserInterface, 49
- MoveInfo
  - scrabble.MoveInfo, 25
- parseInput
  - scrabble.UserInput, 47
- placeTile
  - scrabble.Board, 8
- placeTiles
  - scrabble.Board, 8
- Player
  - scrabble.Player, 29
- playerMove
  - scrabble.Scrabble, 37
- playerReset
  - scrabble.Player, 31
- Pool
  - scrabble.Pool, 32
- poolFill
  - scrabble.Pool, 33
- receiveTile
  - scrabble.Pool, 33
- removeMove
  - scrabble.Board, 9
- removeTile
  - scrabble.Frame, 15
  - scrabble.Pool, 33
- removeTiles
  - scrabble.Frame, 16
- resetBoard
  - scrabble.Board, 9
- returnFrame
  - scrabble.Frame, 16
- Scrabble
  - scrabble.Scrabble, 34
- scrabble.Board, 5
  - BINGO, 10
  - Board, 6
  - BOARD\_SIZE, 10
  - calculateScore, 6
  - checkPlayerHasTiles, 6
  - checkValidMove, 7
  - checkValidPosition, 7
  - getBoardSquares, 7
  - getSquare, 8
  - placeTile, 8
  - placeTiles, 8
  - removeMove, 9
  - resetBoard, 9
  - setWordSquaresNormal, 9
  - toString, 9
- scrabble.BoardTest, 10
- scrabble.exceptions.InvalidBoardException, 18
- scrabble.exceptions.InvalidFrameException, 19
- scrabble.exceptions.InvalidInputException, 19
- scrabble.exceptions.InvalidMoveInfoException, 20
- scrabble.exceptions.InvalidPlayerNameException, 20
- scrabble.exceptions.InvalidPlayerScoreException, 21
- scrabble.exceptions.InvalidPoolException, 21
- scrabble.exceptions.InvalidScrabbleException, 22
- scrabble.exceptions.InvalidSquareException, 22
- scrabble.exceptions.InvalidTileException, 23
- scrabble.exceptions.InvalidWordException, 23
- scrabble.Frame, 11
  - addTile, 12
  - checkTiles, 12
  - fillFrame, 13
  - Frame, 11
  - FRAME\_SIZE, 18
  - getTile, 13
  - getTiles, 14
  - hasBlank, 14
  - isEmpty, 14
  - removeTile, 15
  - removeTiles, 16
  - returnFrame, 16
  - setBlanks, 17
  - setToBlank, 17
  - swapTiles, 17
  - tileValues, 17
  - toString, 18
- scrabble.Main, 24
- scrabble.MoveInfo, 24
  - addAuxiliaryWord, 26
  - getAuxiliaryWords, 26
  - getMoveScore, 26
  - getPlayer, 26
  - getPrimaryWord, 27
  - getRequiredTiles, 27
  - getRequiredTilesPositions, 27
  - MoveInfo, 25
  - setRequiredTiles, 27
  - setScore, 28
- scrabble.MoveInfoTest, 28
- scrabble.Player, 28
  - decreaseScore, 29
  - getName, 30
  - getPlayerFrame, 30
  - getScore, 30
  - increaseScore, 30
  - Player, 29
  - playerReset, 31
  - setName, 31
  - toString, 31

- scrabble.Pool, 32
  - isEmpty, 32
  - Pool, 32
  - poolFill, 33
  - receiveTile, 33
  - removeTile, 33
  - tilesInPool, 33
  - toString, 34
- scrabble.PoolTest, 34
- scrabble.Scrabble, 34
  - createPlayer, 35
  - dictionaryWords, 35
  - gameOver, 35
  - getBoard, 36
  - getMoveHistory, 36
  - getPlayers, 36
  - getPool, 36
  - isGameOver, 36
  - playerMove, 37
  - Scrabble, 34
- scrabble.ScrabbleTest, 37
- scrabble.Square, 37
  - getTile, 38
  - getType, 38
  - isEmpty, 39
  - setEmpty, 39
  - setNormal, 39
  - setTile, 39
  - Square, 38
  - toString, 40
- scrabble.Square.SquareType, 40
- scrabble.SquareTest, 40
- scrabble.Tile, 41
  - compareTo, 42
  - equals, 42
  - getCharacter, 42
  - getValue, 43
  - main, 43
  - setCharacter, 43
  - setNull, 44
  - Tile, 41
  - toString, 44
- scrabble.UserInput, 44
  - getInputType, 46
  - getName, 46
  - getStartPosition, 46
  - getWord, 47
  - getWordDirection, 47
  - parseInput, 47
  - UserInput, 45, 46
- scrabble.UserInput.Direction, 10
- scrabble.UserInput.UserInputType, 48
- scrabble.UserInputTest, 48
- scrabble.userInterface.UserInterface, 48
  - main, 49
  - start, 49
- scrabble.Word, 49
  - getDirection, 50
  - getStartPosition, 50
  - getWord, 51
  - toString, 51
  - Word, 50
- scrabble.WordTest, 51
- setBlanks
  - scrabble.Frame, 17
- setCharacter
  - scrabble.Tile, 43
- setEmpty
  - scrabble.Square, 39
- setName
  - scrabble.Player, 31
- setNormal
  - scrabble.Square, 39
- setNull
  - scrabble.Tile, 44
- setRequiredTiles
  - scrabble.MoveInfo, 27
- setScore
  - scrabble.MoveInfo, 28
- setTile
  - scrabble.Square, 39
- setToBlank
  - scrabble.Frame, 17
- setWordSquaresNormal
  - scrabble.Board, 9
- Square
  - scrabble.Square, 38
- start
  - scrabble.userInterface.UserInterface, 49
- swapTiles
  - scrabble.Frame, 17
- Tile
  - scrabble.Tile, 41
- tilesInPool
  - scrabble.Pool, 33
- tileValues
  - scrabble.Frame, 17
- toString
  - scrabble.Board, 9
  - scrabble.Frame, 18
  - scrabble.Player, 31
  - scrabble.Pool, 34
  - scrabble.Square, 40
  - scrabble.Tile, 44
  - scrabble.Word, 51
- UserInput
  - scrabble.UserInput, 45, 46
- Word
  - scrabble.Word, 50